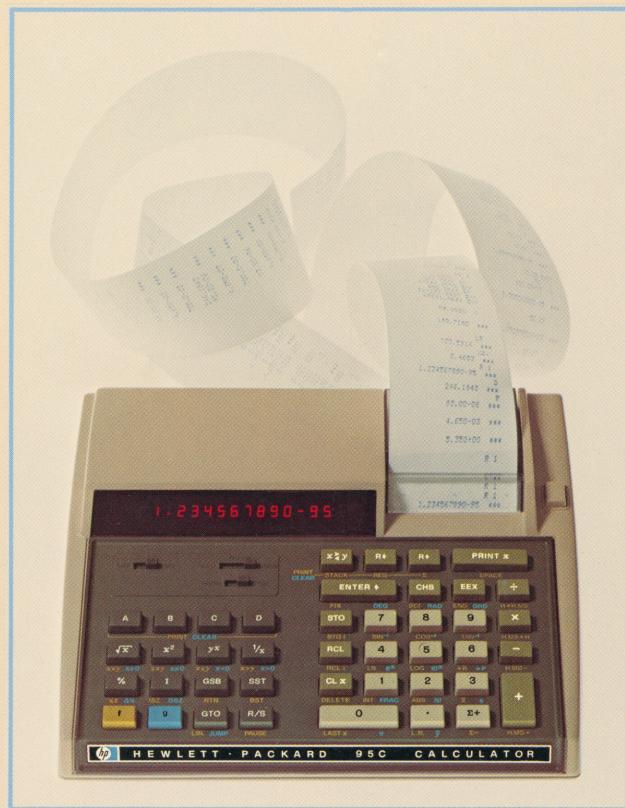


HEWLETT-PACKARD  
**HP·95C**

---



---

**APPLICATIONS**

---

The program material contained herein is supplied without representation or warranty of any kind. Hewlett-Packard Company therefore assumes no responsibility and shall have no liability, consequential or otherwise, of any kind arising from the use of this program material or any part thereof.

## INTRODUCTION

Howdy! This is *your* HP-95C Applications Book, your introduction to real world programs written in a number of fields including mathematics, statistics, finance, surveying, navigation and games. These programs demonstrate the diversity of uses for the HP-95C, and will give you immediate calculation aids for problems you encounter every day. You will also find the programs useful as guides to programming techniques and models for writing your own customized software. The comments accompanying each program will demonstrate the approach used to reach the solution and will help you follow what the programmer was thinking as you become an expert with your HP-95C.

Whether your interest lies in solving a particular problem in a specific area, or in learning more about the programming power of your calculator, we hope that this book will help you get the most from your HP-95C.

## TABLE OF CONTENTS

<i>Introduction</i> .....	1
<i>A Word About Program Usage</i> .....	4
 <b>Algebra and Number Theory</b>	
Quadratic Equation .....	6
Complex Operations .....	8
$3 \times 3$ Matrix Operations .....	12
Base Conversions .....	17
Primes .....	20
Factors .....	21
Greatest Common Divisor, Least Common Multiple .....	23
 <b>Finance</b>	
Annuities and Compound Amounts .....	26
 <b>Games and Such</b>	
Moon Rocket Lander .....	36
Submarine Hunt .....	39
Biorhythms .....	48
Count-Down Timer .....	51
Calender Functions .....	53
 <b>Navigation</b>	
Rhumbline Sequence .....	58
Great Circle Navigation .....	63
Sight Reduction Table .....	65
Time of Sunrise and Sunset .....	67
 <b>Numerical Methods</b>	
Solution to $f(x) = 0$ .....	70
Numerical Integration, Discrete Case .....	72
Numerical Integration, Simpson's Rule .....	75
Differential Equations .....	77
 <b>Statistics</b>	
Curve Fitting .....	80
Covariance and Correlation Coefficient .....	86
Moments and Skewness .....	88
Normal Distribution .....	90
Inverse Normal Integral .....	92
Permutation and Combination .....	94
Random Number Generator .....	96
Chi-Square Evaluation .....	98
Paired t Statistic .....	101
t Statistic for Two Means .....	103

**Structures**

Static Equilibrium of a Point .....	106
Composite Section Properties .....	110
Cantilever Beams—Point Load .....	119
Simply Supported Beams—Point Load .....	123
Fixed-Fixed Beam—Point Load .....	126

**Surveying**

Field Angle Traverse .....	132
Bearing Traverse .....	135
Inverse .....	138
Sideshots .....	141

**Trigonometry and Analytical Geometry**

Coordinate Translation and Rotation .....	144
Triangle Solutions .....	147
Vector Operations .....	153
Hyperbolics .....	157

## A WORD ABOUT PROGRAM USAGE

This HP-95C Applications Book provides a description of the program, a set of instructions for using the program, and one or more example problems, each of which includes a list of the actual keystrokes required for its solution. Explanatory comments have been incorporated in each program listing to facilitate your understanding of the actual working of each program. Thorough study of a commented listing can help you to expand your programming repertoire since interesting techniques can often be found in this way.

The lists of keystrokes required to solve example problems indicate the resulting outputs. Those outputs indicated by \*\*\* are printed with the printer in MANUAL mode. Other outputs appear in the calculator display.

The completed User Instruction Form—which accompanies each program—is your guide to operating the programs in this Pac.

The form is composed of five labeled columns. Reading from left to right, the first column, labeled STEP, gives the instruction step number.

The INSTRUCTIONS column gives instructions and comments concerning the operations to be performed.

The INPUT-DATA/UNITS column specifies the input data, and the units of data if applicable. Data input keys consist of 0 to 9 and decimal point (the numeric keys), EEX (enter exponent), and CHS (change sign).

The KEYS column specifies the keys to be pressed after keying in the corresponding input data.

The OUTPUT-DATA/UNITS column specifies intermediate and final outputs and their units, where applicable.

The following illustrates the User Instruction Form for Quadratic Equation, the first program in this Book.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in coefficients and display D.	c	ENTER+	
		b	ENTER+	
		a	A	(D)
3	If D ≥ 0, roots are real.			x <sub>1</sub>
				x <sub>2</sub>

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
4	If $D < 0$ , roots are complex of form $u \pm iv$ .			u
				v

Step 1 requires you to key in the program. For Quadratic Equation, the program steps are labeled with “A” and an associated step number. To key this program for execution under A, switch the HP-95C to W/PRGM mode, press **9** CLEAR **A**, and key the steps in as listed. The choice of keying in the program under A is arbitrary, and may be changed to fit the user’s needs, making the corresponding changes in the User Instructions shown on the previous page. Note also that some steps on the program listing require keystrokes not explicitly listed for entry in the program, e.g. A-024 LBL 1 is keyed in by three keystrokes **L** **G**<sub>T</sub>**O** **1**. (See the Owner’s Handbook for a more detailed explanation for keying in programs).

Step 2 asks for the coefficients of the quadratic equation. Coefficient c is keyed in and followed by **ENTER**, coefficient b is keyed in and followed by **ENTER**, and coefficient a is keyed in and followed by the **A** key. The discriminant D is calculated and displayed with a pause. The calculator then determines if D is positive or negative, and automatically continues to step 3 or 4, respectively, and prints the roots of the quadratic equation.

# ALGEBRA AND NUMBER THEORY

## QUADRATIC EQUATION

The roots  $x_1, x_2$  of

$$ax^2 + bx + c = 0$$

are given by

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

If

$$D = (b^2 - 4ac)/4a^2$$

is positive or zero, the roots are real. In these cases, better accuracy may sometimes be obtained by first computing the root with the larger absolute value:

If  $-\frac{b}{2a} \geq 0$ ,  $x_1 = -\frac{b}{2a} + \sqrt{D}$

If  $-\frac{b}{2a} < 0$ ,  $x_1 = -\frac{b}{2a} - \sqrt{D}$

In either case,

$$x_2 = \frac{c}{x_1 a} .$$

If  $D < 0$ , the roots are complex, being

$$u \pm iv = \frac{-b}{2a} \pm \frac{\sqrt{4ac - b^2}}{2a} i$$

A-000 LBLA A-001 ENT1 A-002 R↓ A-003 ÷ A-004 × A-005 ÷ A-006 CHS A-007 ENT↑ A-008 X² A-009 R↓ A-010 R↑ A-011 X×Y A-012 ÷ A-013 S 0 A-014 - A-015 PSE A-016 X<0 A-017 GTO3 A-018 JX A-019 X×Y A-020 X<0	$-\frac{b}{2a}$ $b^2/4a^2$ c/a D If $D < 0$ , roots are complex. If $-b/2a < 0$ , GTO 1	A-021 GTO1 A-022 + A-023 GTO2 A-024 LBL1 A-025 X×Y A-026 - A-027 LBL2 A-028 PRTX A-029 1/X A-030 R 0 A-031 X A-032 PRTX A-033 SPC A-034 RTN A-035 LBL3 A-036 CHS A-037 JX A-038 X×Y A-039 PRTX A-040 X×Y A-041 PRTX A-042 SPC	$x_1 = -\frac{b}{2a} + \sqrt{D}$ $x_1 = -\frac{b}{2a} - \sqrt{D}$ $x_1$ $x_2 = \frac{c}{x_1 a}$ Complex roots. $v = \sqrt{-D}$ u v
REGISTERS			

0 c/a	1	2	3
4	5	6	7

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in coefficients and display D.	c	<b>ENTER</b>	
		b	<b>ENTER</b>	
		a	<b>A</b>	(D)
3	If $D \geq 0$ , roots are real.			$x_1$
				$x_2$
4	If $D < 0$ , roots are complex of form $u \pm iv$ .			u
				v

### **Example 1:**

Find the roots of  $x^2 + x - 6 = 0$ .

## Keystrokes:

6 CHS ENTER ↴ 1 ENTER ↴ 1 A →

## Outputs:

6.25 (D)  
-3.00 \*\*\* ( $x_1$ )  
2.00 \*\*\* ( $x_2$ )

### **Example 2:**

Solve the quadratic equation  $2x^2 - 3x + 5 = 0$ .

## Keystrokes:

5 **ENTER** 3 **CHS** **ENTER** 2 **A**

### **Outputs:**

-1.94 (D)  
0.75 \*\*\* (u)  
1.39 \*\*\* (V)

Thus  $x_{1,2} = 0.75 \pm 1.39i$ .

## COMPLEX OPERATIONS

This program allows for chained calculations involving complex numbers. The four operations of complex arithmetic ( $+$ ,  $-$ ,  $\times$ ,  $\div$ ) are provided, as well as several of the most used functions of a complex variable  $z$  ( $1/z$ ,  $z^n$ , and  $z^{1/n}$ ). Functions and operations may be mixed in the course of a calculation to allow evaluation of expressions like  $z_3/(z_1 + z_2)$ ,  $z^{-2}$ ,  $z_1z_2^3$ , etc., where  $z_1$ ,  $z_2$ ,  $z_3$  are complex numbers of the form  $a + ib$ .

### **Equations:**

$$\text{Let } z_k = a_k + ib_k = r_k e^{i\theta_k}, \quad k = 1, 2$$

Let the result in each case be  $u + iv$ .

$$z_1 + z_2 = (a_1 + a_2) + i(b_1 + b_2)$$

$$z_1 - z_2 = (a_1 - a_2) + i(b_1 - b_2)$$

$$z_1 z_2 = r_1 r_2 e^{i(\theta_1 + \theta_2)}$$

$$z_1/z_2 = \frac{r_1}{r_2} e^{i(\theta_1 - \theta_2)}$$

$$1/z = \frac{1}{r} e^{-i\theta}, \quad z \neq 0$$

$$z^n = r^n e^{in\theta} \quad (n = \text{integer})$$

$$z^{1/n} = r^{1/n} e^{i\left(\frac{\theta}{n} + \frac{360k}{n}\right)}, \quad k = 0, 1, \dots, n-1 \quad (n = \text{integer})$$

(All  $n$  roots will be output and temporarily stored during the calculation of  $z^{1/n}$ ,  $k = 0, 1, \dots, n-1$ ; at the end of the calculation, the final root will be stored.)

### **Remarks:**

The logic system for this program may be thought of as a kind of Reverse Polish Notation (RPN) with a stack whose capacity is two complex numbers. Let the bottom register of the complex stack be  $\xi$  and the top register  $\tau$ . These are analogous to the X, Y- and Z, T- registers in the calculator's own four-register stack.\* A complex number  $z_1$  is input to the  $\xi$ -register by the key-strokes  $a_1 \text{ [ENTER]} b_1 \text{ [A]}$ . Upon input of a second complex number  $z_2$  (as  $a_2 \text{ [ENTER]} b_2 \text{ [A]}$ ),  $z_1$  is moved to  $\tau$  and  $z_2$  is placed in  $\xi$ . The previous contents of  $\tau$  are lost.

Functions operate on the  $\xi$ -register, and the result is left in  $\xi$ ;  $\tau$  is unchanged. Arithmetic operations involve both the  $\xi$ - and  $\tau$ - registers; and result of the

operation is left in  $\xi$  and  $\tau$  is unchanged.

\*Each register of the complex stack must actually hold two real numbers: the real and the imaginary part of its complex contents. Thus, it takes two of the calculator's registers to represent one register in the complex stack. In this discussion we speak of the complex stack registers as though they were each just one register.

<b>A-000 LBL4</b>	Input a↑b	<b>A-057 +</b>	
<b>A-001 R 3</b>	Last b → R <sub>1</sub> (b <sub>1</sub> )	<b>A-058 X↓Y</b>	r <sub>1</sub> r <sub>2</sub> (θ <sub>1</sub> + θ <sub>2</sub> )
<b>A-002 S 1</b>		<b>A-059 R↑</b>	-----
<b>A-003 R↓</b>		<b>A-060 X</b>	Output routine
<b>A-004 S 3</b>	Present b → R <sub>3</sub> (b <sub>2</sub> )	<b>A-061 LBL9</b>	
<b>A-005 R↓</b>	Last a → R <sub>0</sub> (a <sub>1</sub> )	<b>A-062 +R</b>	
<b>A-006 R 2</b>		<b>A-063 S 2</b>	
<b>A-007 S 0</b>	Present a → R <sub>2</sub> (a <sub>2</sub> )	<b>A-064 PRTX</b>	
<b>A-008 R↓</b>		<b>A-065 X↓Y</b>	
<b>A-009 S 2</b>	-----	<b>A-066 S 3</b>	
<b>A-010 RTN</b>		<b>A-067 PRTX</b>	
<b>A-011 LBL1</b>	Add (+)	<b>A-068 SPC</b>	
<b>A-012 R 0</b>		<b>A-069 RTN</b>	-----
<b>A-013 S+2</b>		<b>A-070 LBL5</b>	1/z
<b>A-014 R 2</b>		<b>A-071 R 3</b>	
<b>A-015 PRTX</b>	a <sub>2</sub> ← a <sub>1</sub> + a <sub>2</sub>	<b>A-072 R 2</b>	
<b>A-016 R 1</b>		<b>A-073 +P</b>	r θ
<b>A-017 S+3</b>		<b>A-074 X↓Y</b>	
<b>A-018 R 3</b>	b <sub>2</sub> ← b <sub>1</sub> + b <sub>2</sub>	<b>A-075 CHS</b>	
<b>A-019 FRTX</b>	-----	<b>A-076 X↓Y</b>	
<b>A-020 SPC</b>		<b>A-077 1/X</b>	1/r - θ
<b>A-021 RTN</b>		<b>A-078 GT09</b>	-----
<b>A-022 LBL2</b>	Subtract (-)	<b>A-079 LBL6</b>	
<b>A-023 R 0</b>		<b>A-080 S 1</b>	
<b>A-024 R 2</b>		<b>A-081 R 3</b>	
<b>A-025 -</b>		<b>A-082 R 2</b>	
<b>A-026 S 2</b>	a <sub>2</sub> ← a <sub>1</sub> - a <sub>2</sub>	<b>A-083 +P</b>	r θ
<b>A-027 PRTX</b>		<b>A-084 I</b>	
<b>A-028 R 1</b>	b <sub>2</sub> ← b <sub>1</sub> - b <sub>2</sub>	<b>A-085 Y<sup>x</sup></b>	
<b>A-029 R 3</b>		<b>A-086 X↓Y</b>	
<b>A-030 -</b>		<b>A-087 I</b>	
<b>A-031 S 3</b>		<b>A-088 X</b>	
<b>A-032 PRTX</b>		<b>A-089 X↓Y</b>	r <sup>n</sup> nθ
<b>A-033 SPC</b>	-----	<b>A-090 GT09</b>	-----
<b>A-034 RTN</b>		<b>A-091 LBL7</b>	
<b>A-035 LBL3</b>	Multiply (x)	<b>A-092 S 1</b>	
<b>A-036 R 1</b>		<b>A-093 1</b>	
<b>A-037 R 0</b>		<b>A-094 CHS</b>	
<b>A-038 +P</b>	r <sub>1</sub> θ <sub>1</sub>	<b>A-095 COS~</b>	Compute 360/n (or 2π/n, or
<b>A-039 R 3</b>		<b>A-096 ENT↑</b>	400/n)
<b>A-040 R 2</b>		<b>A-097 +</b>	
<b>A-041 +P</b>	r <sub>2</sub> θ <sub>2</sub> r <sub>1</sub> θ <sub>1</sub>	<b>A-098 X↓Y</b>	
<b>A-042 GT00</b>	-----	<b>A-099 ÷</b>	
<b>A-043 LBL4</b>	Divide (÷)	<b>A-100 S 4</b>	
<b>A-044 R 1</b>		<b>A-101 R 3</b>	
<b>A-045 R 0</b>		<b>A-102 R 2</b>	
<b>A-046 +P</b>	r <sub>1</sub> θ <sub>1</sub>	<b>A-103 +P</b>	r θ
<b>A-047 R 3</b>		<b>A-104 I</b>	
<b>A-048 R 2</b>		<b>A-105 1/X</b>	
<b>A-049 +P</b>	r <sub>2</sub> θ <sub>2</sub> r <sub>1</sub> θ <sub>1</sub>	<b>A-106 Y<sup>x</sup></b>	
<b>A-050 X↓Y</b>		<b>A-107 X↓Y</b>	
<b>A-051 CHS</b>		<b>A-108 I</b>	
<b>A-052 X↓Y</b>		<b>A-109 ÷</b>	
<b>A-053 1/X</b>	1/r <sub>2</sub> - θ <sub>2</sub> r <sub>1</sub> θ <sub>1</sub>	<b>A-110 X↓Y</b>	r <sup>1/n</sup> θ/n
<b>A-054 LBL6</b>	-----	<b>A-111 LBL8</b>	-----
<b>A-055 X↓Y</b>		<b>A-112 GS9</b>	Convert → R and print
<b>A-056 R↑</b>		<b>A-113 D5Z</b>	

A-114 GT00 A-115 RTW A-116 LBLO A-117 X $\leftrightarrow$ Y A-118 $\rightarrow$ P	Loop n times Back $\rightarrow$ P ( $r^{1/n}$ , $\theta/n$ )	A-119 X $\leftrightarrow$ Y A-120 R 4 A-121 + A-122 X $\leftrightarrow$ Y A-123 GT08	$\theta/n + \frac{360}{n} k$
<b>REGISTERS</b>			
0 $a_1$	1 $b_1$	2 $a_2$	3 $b_2$
4 360/n	5	6	7
8	9	.0	.1
.2	.3	.4	.5
			I $n$

STEP	INSTRUCTIONS	INPUTS DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in first complex number, $a_1 + ib_1$ .	$a_1$	<b>ENTER</b>	
		$b_1$	<b>A</b>	$a_1$
3	For arithmetic, go to step 4; for a function, go to step 7. A complex result is $u + iv$ .			
	<b>Arithmetic</b>			
4	Key in second complex number, $a_2 + ib_2$ .	$a_2$	<b>ENTER</b>	
		$b_2$	<b>A</b>	$a_2$
5	Select one of four operations: • Add (+)		<b>GSB [1]</b>	$u$
				$v$
	• Subtract (-)		<b>GSB [2]</b>	$u$
				$v$
	• Multiply (x)		<b>GSB [3]</b>	$u$
				$v$
	• Divide (÷)		<b>GSB [4]</b>	$u$
				$v$
6	The result of the operation has been stored; go to step 7 for a			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	function or to step 4 for further arithmetic.			
	<b>Functions</b>			
7	Select one of three functions:			
	• Reciprocal ( $1/z$ )		GSB [5]	u
				v
	• Raise z to an integer power ( $z^n$ )	n	GSB [6]	u
				v
	• Find the $n^{\text{th}}$ root of z ( $z^{1/n}$ )	n	GSB [7]	u
				v
8	The result has been stored; go to step 4 for arithmetic or to step 7 for another function.			

**Example 1:**

Evaluate the expression

$$\frac{z_1}{z_2 + z_3},$$

where  $z_1 = 23 + 13i$ ,  $z_2 = -2 + i$ ,  $z_3 = 4 - 3i$ . Since the program can remember only two numbers at a time, perform the calculation as

$$z_1 \times [1/(z_2 + z_3)].$$

**Keystrokes:**

2 A ENTER↑ 1 A 4 ENTER↑ 3  
 CHS A GSB 1 →  
 GSB [5] →  
 23 ENTER↑ 13 A GSB [3] →

**Outputs:**

2.00 \*\*\* real  $(z_2 + z_3)$   
 -2.00 \*\*\* imag  $(z_2 + z_3)$   
 0.25 \*\*\*  $(1/(z_2 + z_3))$   
 0.25 \*\*\*  
 2.50 \*\*\*  $(z_1/(z_2 + z_3))$   
 9.00 \*\*\*

**Example 2:**

Find the 3 cube roots of 8.

**Keystrokes:**

8 **ENTER** 0 **A** 3 **GSB** **7** —————→

**Outputs:**

2.00	***
0.00	***
-1.00	***
1.73	***
-1.00	***
-1.73	***

**3 × 3 MATRIX OPERATIONS**

This program performs some of the most common operations involving  $3 \times 3$  matrices; namely, the calculation of the determinant and inverse of the matrix and the solution of a system of three simultaneous equations in three unknowns.

The method used in this program is Gaussian elimination. No pivoting is performed. There is one restriction on the matrix values, i.e., that the upper left-hand element ( $a_1$ ) not be zero. If  $a_1 = 0$ , interchange the first row (or column) with another row (or column). This interchange will result in (1) a sign change in the determinant and (2) the interchange of the corresponding *columns* (or rows) in the inverse matrix.

Space does not permit a full treatment of the pertinent equations; however, the Comments section of the program listing shows the operations in detail. Basically, the input matrix A is transformed into an upper triangular matrix U, assuming A is nonsingular. The multipliers used to accomplish this transformation form a lower triangular matrix L, which has 1's along its diagonal. Disregarding one possible permutation matrix, the relationship among these matrices is  $U = LA$ . The matrix A is destroyed and the matrices U and L are used to compute the determinant, inverse, and solutions of simultaneous linear equations.

**Equations:**

$$\text{Let } A = \begin{matrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{matrix}, a_1 \neq 0.$$

The determinant of A, Det A, is found *after* its transformation to U by the product of the diagonal elements:

$$\text{Det } A = \pm a_1 b_2 c_3.$$

The sign is negative if a row interchange was performed by the program in the transformation of A to U.

A set of 3 simultaneous equations in 3 unknowns may be written as

$$a_1 x + b_1 y + c_1 z = d_1$$

$$a_2 x + b_2 y + c_2 z = d_2$$

$$a_3 x + b_3 y + c_3 z = d_3$$

where x, y, z are unknowns and the  $\{d_i\}$  constants.

In matrix notation this becomes  $Ax = d$ , where x and d are the column vectors

$$\begin{array}{ll} x & d_1 \\ y & \text{and} & d_2 \\ z & & d_3 \end{array}, \text{ respectively.}$$

This problem is solved (neglecting row interchanges) as  $Ux = Ld$ .

Let T be the inverse of A; i.e., the  $3 \times 3$  matrix is such that  $AT = TA = I$ , the  $3 \times 3$  identity matrix.

$$T = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{pmatrix} \quad I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Let the column vectors of T be denoted  $t^{(j)}$  and let  $I^{(j)}$  be the column vectors of the identity matrix. Then the inverse matrix T is found by the solution of the equations

$$At^{(j)} = I^{(j)}, j = 1, 2, 3.$$

### Remarks:

1. This program may also be used to solve  $2 \times 2$  systems by representing the  $2 \times 2$  matrix  $\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}$  as a  $3 \times 3$  matrix

$$\begin{matrix} a_1 & b_1 & 0 \\ a_2 & b_2 & 0 \\ 0 & 0 & 1 \end{matrix}$$

and a 2-dimensional column vector  $\begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$  as a 3-dimensional column vector  $\begin{pmatrix} d_1 \\ d_2 \\ 0 \end{pmatrix}$ .

2. A determinant of 0 or an error halt in the execution of the program under LBL A with the message "See 5" indicates that the matrix input is singular.

A-000 LBLA		A-051 R 4	
A-001 S 2	Input	A-052 R 3	
A-002 R↓		A-053 S 4	
A-003 S 1		A-054 R↓	
A-004 R↓		A-055 S 3	
A-005 S.1		A-056LBL0	-----
A-006 R/S		A-057 CHS	
A-007 S 4		A-058 S÷4	$m_{32} \leftarrow -\frac{b_3}{b_2}$
A-008 R↓		A-059 R 4	
A-009 S 3		A-060 R 5	
A-010 R↓		A-061 X	
A-011 S.2		A-062 S+6	
A-012 R/S		A-063 R.1	
A-013 S 6		A-064 R 3	
A-014 R↓		A-065 X	
A-015 S 5		A-066 R 6	
A-016 R↓		A-067 X	
A-017 S.3	-----	A-068 R.0	
A-018 1		A-069 X	
A-019 S.0		B-000LBLB	-----
A-020 R.1		B-001 1	Inverse.
A-021 CHS	$m_{11} \leftarrow -\frac{a_1}{a_1}, i = 2, 3$	B-002 ENT↑	
A-022 S=1		B-003 0	
A-023 S÷2		B-004 ENT↑	
A-024 R 1		B-005 C	
A-025 R.2		B-006 0	
A-026 X		B-007 ENT↑	
A-027 S+3	$b_2 \leftarrow b_2 + m_{21} b_1$	B-008 1	
A-028 R 1		B-009 ENT↑	
A-029 R.3		B-010 0	
A-030 X		B-011 C	
A-031 S+5	$c_2 \leftarrow c_2 + m_{21} c_1$	B-012 0	
A-032 R 2		B-013 ENT↑	
A-033 R.2		B-014 ENT↑	
A-034 X	$b_3 \leftarrow b_3 + m_{31} b_1$	B-015 1	
A-035 S+4		B-016 C	
A-036 R 2		C-000LBLC	-----
A-037 R.3		C-001 S 9	Solve Ax = d.
A-038 X		C-002 R↓	
A-039 S+6	$c_3 \leftarrow c_3 + m_{31} c_1$	C-003 S 8	
A-040 R 3		C-004 R↓	
A-041 X≠0	-----	C-005 S 7	
A-042 GT00	If $b_2 = 0$ , swap:	C-006 R 1	
A-043 1		C-007 R 7	Ld
A-044 CHS		C-008 X	
A-045 S.0	$b_2 \leftarrow b_3, c_2 \leftarrow c_3$ .	C-009 S+6	
A-046 R 5		C-010 R 2	
A-047 R 6	Set pivot to -1.	C-011 R 7	
A-048 S 5		C-012 X	
A-049 R↓		C-013 S+9	
A-050 S 6		C-014 R.0	

C-015 X>0 C-016 GT00 C-017 R 8 C-018 R 9 C-019 S 8 C-020 R↓ C-021 S 9 C-022 LBL0 C-023 R 4 C-024 R 8 C-025 x C-026 S+9 C-027 R 6 C-028 S÷9 C-029 R 9 C-030 R 5 C-031 x C-032 S-8 C-033 R 3	If Pivot = -1, swap:  d <sub>2</sub> ≈ d <sub>3</sub> .  -----  d <sub>3</sub> ← d <sub>3</sub> + m <sub>32</sub> d <sub>2</sub> .  z = d <sub>3</sub> /c <sub>3</sub> .	C-034 S÷8 C-035 R 9 C-036 R.3 C-037 x C-038 R 8 C-039 R.2 C-040 x C-041 + C-042 S-7 C-043 R.1 C-044 S÷7 C-045 R 7 C-046 PRTX C-047 R 8 C-048 PRTX C-049 R 9 C-050 PRTX C-051 SPC	y = (d <sub>2</sub> - c <sub>2</sub> z)/b <sub>2</sub>  x = (d <sub>1</sub> - b <sub>1</sub> y - c <sub>1</sub> z)/a <sub>1</sub> .  Output x, y, z.
<b>REGISTERS</b>			
0	1 a <sub>2</sub> , m <sub>21</sub>	2 a <sub>3</sub> , m <sub>31</sub>	3 b <sub>2</sub>
4 b <sub>3</sub> , m <sub>32</sub>	5 c <sub>2</sub>	6 c <sub>3</sub>	7 d <sub>1</sub> , x
8 d <sub>2</sub> , y	9 d <sub>3</sub> , z	.0 ±1 - Pivot?	.1 a <sub>1</sub>
.2 b <sub>1</sub>	.3 c <sub>1</sub>	.4	.5
			I

STEP	INSTRUCTIONS	INPUTS DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in matrix by columns and find determinant.	a <sub>1</sub> a <sub>2</sub> a <sub>3</sub> b <sub>1</sub> b <sub>2</sub> b <sub>3</sub> c <sub>1</sub> c <sub>2</sub> c <sub>3</sub>	ENTER+ ENTER+ A ENTER+ ENTER+ R/S ENTER+ ENTER+ R/S	a <sub>1</sub> b <sub>1</sub> Det A
				t <sub>11</sub>
				t <sub>21</sub>
				t <sub>31</sub>
3	Find inverse by columns.		B	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
				$t_{12}$
				$t_{22}$
				$t_{32}$
				$t_{13}$
				$t_{23}$
				$t_{33}$
4	To solve 3 simultaneous equations, key in the constants and find the unknowns.	$d_1$	<b>ENTER</b>	
		$d_2$	<b>ENTER</b>	
		$d_3$	<b>C</b>	$x$
				$y$
				$z$

**Example:**Solve the  $3 \times 3$  system

$$\begin{aligned}x + 2y - z &= 4 \\3x - y + z &= 7 \\-2x &\quad + 3z = 8\end{aligned}$$

Also find the determinant and inverse of the coefficient matrix.

**Keystrokes:**

- 1 **ENTER** 3 **ENTER** 2 **CHS** **A**  $\rightarrow$   
 2 **ENTER** 1 **CHS** **ENTER** 0 **R/S**  $\rightarrow$   
 1 **CHS** **ENTER** 1 **ENTER**  
 3 **R/S**  $\longrightarrow$   $\rightarrow$

**Outputs:**

- 1.00  
 2.00  
 -23.00 (Det A)  
**B**  $\longrightarrow$   $\rightarrow$   
 0.13 \*\*\* (Inverse)  
 0.48 \*\*\*  
 0.09 \*\*\*  
 0.26 \*\*\*

	-0.04 ***
	0.17 ***
	-0.04 ***
	0.17 ***
	0.30 ***
4 <b>ENTER</b> 7 <b>ENTER</b> 8 <b>C</b> →	2.00 *** (x) 3.00 *** (y) 4.00 *** (z)

## BASE CONVERSIONS

This program converts positive numbers to and from base 10 representations. The other base involved may be any integer from 2 to 99, inclusive.

Let  $x_b$  be the representation of the number in the original base  $b$ . Assume that it is to be converted to the representation  $x_B$  in base  $B$ . Either  $b$  or  $B$  must be 10. In general, the bases are stored manually ( $b$  in  $R_1$ ,  $B$  in  $R_2$ ) prior to keying in  $x_b$  and pressing **A**, which will cause the computation of  $x_B$ .

When converting numbers from base 10,  $b = 10$ . However, the number stored for  $b$  may be either 10 or 100. If the other base  $B < 10$ , then store  $b$  in  $R_1$  as 10. If, however,  $B > 10$ , the value stored for  $b$  in  $R_1$  should be 100.

Similarly, when numbers are converted to base 10 representations,  $B = 10$ . When  $b < 10$ , the value of  $B$  stored in  $R_2$  should be 10; when  $b > 10$ , a value of 100 should be stored in  $R_2$ .

The table below shows examples of the four possible cases:

To convert	From Base	To Base	Store in $R_1$	Store in $R_2$
10	2	10	10	2
10	16	100	100	16
2	10	2	2	10
16	10	16	16	100

A number such as  $4B6_{16}$  cannot be represented directly on the display because the display is strictly numeric. Therefore, some convention must be adopted to represent numbers  $R_a$  when  $a > 10$ . We use the convention of allocating two digit locations for each single character in  $R_a$  when  $a > 10$ .

For example,  $4B6_{16}$  is represented as  $041106_{16}$  by our convention (in hexadecimal system, A = 10, B = 11, C = 12, D = 13, E = 14, F = 15).

When displayed, this number may appear as 41106 or with an exponent

4.1106      04

which is interpreted as  $4.B6 \times 16^2$ .

The displayed exponent 4 is for base 10 and only serves to locate the decimal point (in the same manner as for decimal numbers).

When base  $a > 10$  (as in the above example), divide the displayed exponent by 2 to get the true exponent of the number. When the displayed exponent is an odd integer, shift the decimal point of the displayed number one place (to the left or right) and adjust its exponent accordingly to make the true exponent an integer.

For example, the displayed number

1.112            -03

is interpreted as  $B.C \times 16^{-2}$  or  $0.BC \times 16^{-1}$ .

### Remarks:

1. When the magnitude of the number is very large or very small, this program will take a long time to execute.
2. The program will not give an error indication for invalid inputs for  $x_b$ . For example,  $981_8$  will be treated the same as  $1201_8$ .
3. As the program now stands, the user is forced to make a decision at input time whether the number stored for base 10 is 10 or 100. An alternative approach would be to always store 10, never 100, and have the program decide whether to overwrite the 10 with 100 in some cases. Such an alteration of the program would require about 25 more program steps.

This program may generate roundoff error in the least significant digits for some problems.

A-000 LBLA	$x_b \rightarrow x_B$ .	A-024 LBL8	On entry $R_3$ contains normalized
A-001 S 3		A-025 R 6	
A-002 R 1		A-026 R 3	
A-003 S 5		A-027 x	
A-004 R 2		A-028 S 3	
A-005 S 6		A-029 GSB7	
A-006 C		A-030 R 4	
A-007 S 0		A-031 R 5	
A-008 S 4		A-032 x	
A-009 EEX		A-033 +	
A-010 1		A-034 S 4	
A-011 2		A-035 R 3	
A-012 S 8		A-036 GSB7	
A-013 R 3		A-037 R 3	
A-014 LBL9	-----	A-038 -	
A-015 1	Shift right until < 1.	A-039 ABS	
A-016 X>Y	R <sub>0</sub> keeps track of no. places shifted (exponent)..	A-040 S 3	Do not build mantissa beyond $10^{12}$ .
A-017 GT08		A-041 1	
A-018 S+0		A-042 S-0	
A-019 CL x		A-043 R 4	
A-020 R 6		A-044 R 8	
A-021 ÷		A-045 X≤Y	
A-022 S 3	-----	A-046 GT03	
A-023 GT09	-----	A-047 R 3	-----

A-048 X#0 A-049 GT08 A-050 LBL3 A-051 R 5 A-052 R 0 A-053 YX A-054 R 4 A-055 X A-056 S 4		A-057 RTN A-058 LBL7 A-059 EEX A-060 2 A-061 + A-062 EEX A-063 2 A-064 - A-065 INT	----- Eliminate round-off error -----
<b>REGISTERS</b>			
0 Used	1 b	2 B	3 $x_B$
4 Used	5 b	6 B	7
8 $10^{12}$	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store bases (one must be 10 or 100):			
	• Base to be converted from	b	<b>STO [1]</b>	
	• Base to be converted to	B	<b>STO [2]</b>	
3	Key in number in base b and calculate number in base B.	$x_b$	<b>A</b>	$x_B$
4	For a new conversion between the same bases, go to step 3; to change either base, go to step 2.			

**Example 1:**

Convert  $0.2937_{10}$  to base 8 representation. (Since  $B = 8 < 10$ ,  $b = 10$ .)

**Keystrokes:**10 **STO [1]** 8 **STO [2]** **f** **FIX [9]**.2937 **A**  $\longrightarrow$  0.226277543 (Base 8)**Outputs:****Example 2:**

Convert  $1.23_{10} \times 10^{-12}$  to base 16. (Since  $B = 16 > 10$ ,  $b = 100$ .)

**Keystrokes:**100 **STO [1]** 16 **STO [2]****Outputs:**

1.23 EEX CHS 12 A → 1.051003061–20 (Base 16)

This is interpreted as  $1.5A36A_{16} \times 16^{-10}$ .

### Example 3:

Convert  $7.200067_8 \times 8^{-10}$  to base 10. (Since  $b = 8 < 10$ ,  $B = 10$ .)

**Keystrokes:**

**Outputs:**

8 STO 1 10 STO 2

f SCI 9 7.200067

EEX CHS 10 A → 6.752284066–09 (Base 10)

### Example 4:

Convert  $D.2EE4_{16} \times 16^{12}$  to base 10. (Since  $b = 16 > 10$ ,  $B = 100$ .)

**Keystrokes:**

**Outputs:**

16 STO 1 100 STO 2

13.02141404 EEX 24 A → 3.710731485 15 (Base 10)

## PRIMES

This program will list all prime numbers beginning at a positive integer  $P$  ( $P \geq 3$ ) specified by the user.

The program tests all odd numbers as potential primes and potential divisors. The simplicity of this technique leads to a short program with relatively long execution time. A longer and faster program could be written using an algorithm which tests only a certain subset of odd numbers as potential primes and/or potential divisors. The algorithm might exclude from the test all numbers divisible by 3 or, with an even longer program, all numbers divisible by both 3 and 5.

A-000 LBL A A-001 2 A-002 ÷ A-003 INT A-004 2 A-005 × A-006 1 A-007 + A-008 S 0 A-009 LBL B A-010 3	Select 1 <sup>st</sup> odd no. ≥ input.	A-011 S 1 A-012 LBL B A-013 R 0 A-014 R 1 A-015 ÷ A-016 LST X A-017 X>Y A-018 GTO Ü A-019 X×Y A-020 INT A-021 LST X	If divisor d > n/d, then $d > \sqrt{n}$ ; exit.  If d/n, n is not prime; select new n.
---	--	---	---

A-022 X=Y A-023 GT01 A-024 2 A-025 S+1 A-026 GT09 A-027 LBL0	Else d $\leftarrow$ d + 2; test new d.		A-028 R Ü A-029 PRTX A-030 LBL1 A-031 2 A-032 S+0 A-033 GT08	n is prime; print.  Select next n.
<b>REGISTERS</b>				
0 n	1 d	2	3	
4	5	6	7	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in lower bound of search			
	for primes.	P	A	Primes

**Example:**

List primes greater than 100.

**Keystrokes:**

100 A →

**Outputs:**

101.00 \*\*\*  
103.00 \*\*\*  
107.00 \*\*\*  
109.00 \*\*\*  
113.00 \*\*\*

R/S (Halt program.)

**FACTORS**

This program finds all prime factors of a positive integer.

As in the primes program, all odd numbers and 2 are tested as potential factors. The program could be made faster, though longer, by excluding from the test all potential factors divisible by 3 and 5.

<pre> A-000 LBL4 A-001 S 0 A-002   1 A-003 S 1 A-004   2 A-005 S 2 A-006 LBL7 A-007 R 0 A-008 R 2 A-009 ÷ A-010 LSTX A-011 X&gt;Y A-012 GT00 A-013 X≥Y A-014 INT A-015 LSTX A-016 X=Y </pre>	<p>n is no. to be factored.</p> <p>d in R<sub>2</sub> is trial divisor.</p> <p>If d &gt; n/d, then d &gt; <math>\sqrt{n}</math>; n prime.</p> <p>If n/d integer, d is factor; print.</p>	<pre> A-017 GT01 A-018   2 A-019 S+1 A-020 R 1 A-021 S 2 A-022 GT07 A-023 LBL0 A-024 R 0 A-025 PRTX A-026 RTN A-027 LBL1 A-028 R 2 A-029 PRTX A-030 R↓ A-031 S 0 A-032 GT07 </pre>	<p>Otherwise select next d.</p> <p>n is prime; print as factor.</p> <p>d is a factor.</p> <p><math>n \leftarrow n/d</math>.</p>
<b>REGISTERS</b>			
0      n	1      1, 3, 5, ...	2      d	3
4	5	6	7

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in integer to be factored.	n	A	Factors

**Example 1:**

Find the prime factors of 924.

**Keystrokes:**

924 A →

 2.00 \*\*\*
   
 2.00 \*\*\*
   
 3.00 \*\*\*
   
 7.00 \*\*\*
   
 11.00 \*\*\*
**Outputs:**

$$\text{i.e., } 924 = 2 \times 2 \times 3 \times 7 \times 11.$$

**Example 2:**

Find the prime factors of 3623.

**Keystrokes:**

3623 A →

**Outputs:**

3623.00 \*\*\*

That is, 3623 is prime.

## GREATEST COMMON DIVISOR, LEAST COMMON MULTIPLE

This program will find the greatest common divisor or the least common multiple of two positive integers.

Given two integers  $a$  and  $b$ , their greatest common divisor,  $\text{GCD}(a,b)$ , is found as follows:

1. If  $b = 0$ ,  $\text{GCD}(a,b) \leftarrow a$  and the program will halt.
2. If  $b \neq 0$ ,  $z \leftarrow a \bmod b$ ,  $a \leftarrow b$ , and  $b \leftarrow z$ . Return to 1.

The least common multiple  $\text{LCM}(a,b)$  is found by

$$\text{LCM}(a,b) = \frac{ab}{\text{GCD}(a,b)}$$

A-000 LBL4 A-001 S 3 A-002 X#Y A-003 S 2 A-004 RTN A-005 LBL1 A-006 R 3 A-007 S 1 A-008 R 2 A-009 S 0 A-010LBL5 A-011 R 0 A-012 R 0 A-013 R 1 A-014 S 0 A-015 ÷ A-016 INT	$a \uparrow b$ . ----- $\text{GCD}(a, b)$ $b \rightarrow R_1$ $a \rightarrow R_0$ $z \leftarrow a \bmod b$ $= a - b * \text{INT}(a/b)$ .	A-017 R 1 A-018 x A-019 - A-020 S 1 A-021 X#0 A-022 GT05 A-023 R 0 A-024 RTN A-025LBL2 A-026 R 2 A-027 R 3 A-028 x A-029 S 4 A-030 GSB1 A-031 R 4 A-032 X#Y A-033 ÷	$a \leftarrow b$ . $b \leftarrow z$ . If $b \neq 0$ , loop again. ----- If $b = 0$ , $a = \text{GCD}$ . ----- $R_4 \leftarrow a * b$ . $\text{LCM} = ab/\text{GCD}$ .
<b>REGISTERS</b>			
0 a	1 b	2 a	3 b
4 ab	5	6	7
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in integers.	a	<b>ENTER</b>	
		b	<b>A</b>	a
3	Find either or both:			
	• Greatest common divisor		<b>GSB [1]</b>	GCD (a,b)
	• Least common multiple		<b>GSB [2]</b>	LCM (a,b)

**Example:**

Find the GCD and LCM of 406 and 266.

**Keystrokes:**

406 **ENTER** 266 **A** **GSB** **1** —→  
**GSB** **2** ——————→

**Outputs:**

14.00      (GCD)  
7714.00      (LCM)

## **NOTES**

# FINANCE

## ANNUITIES AND COMPOUND AMOUNTS

This program can be used to solve a variety of problems involving money, time and interest. The following variables can be inputs or outputs:

- n, which is the number of compounding periods. (For a 30 year loan with monthly payments,  $n = 12 \times 30 = 360$ .)
- i, which is the periodic interest rate expressed as a percent. (For other than annual compounding, divide the annual percentage rate by the number of compounding periods in a year; i.e. 8% annual interest compounded monthly equals 8/12 or 0.667%.)
- PMT, which is the periodic payment.
- PV, which is the present value of the cash flows or compound amount.
- FV, which is the future value of a compounded amount or a series of cash flows.
- BAL, which is the balloon or remaining balance at the end of a series of payments.

Accumulated interest and remaining balance may also be computed with this program.

The program accommodates payments which are made at the end of compounding periods or at the beginning. Payments made at the end of compounding periods (ordinary annuity) are common in direct reduction loans and mortgages while payments at the beginning of compounding periods (annuity due) are common in leasing.

This program uses the convention that **cash outlays are input as negative**, and **cash incomes are input as positive**.

The initialization (**A**) performs two functions:

1. It sets PMT, PV, and BAL to zero (n and i are not affected).
2. It sets the ordinary annuity mode.

Pressing **A** provides a safe, convenient, easy to remember method of preparing the calculator for a new problem. It is not necessary to use **A** between problems containing the same combination of variables. For instance, any number of n, i, PMT, FV problems involving different numbers and/or different combinations of knowns could be done in succession without re-initializing. Only the values which change from problem to problem would have to be keyed in. To change the combination of variables without using **A**, simply input zero for any variable which is no longer applicable. To go from n, i, PMT, PV problems to n, i, PV, FV problems, a zero would be stored (0 **STO** **3**) in place of PMT. Table I summarizes these procedures.

**Table I**  
Possible Solutions Using *Annuities and Compound Amounts*

Allowable Combination of Variables	Applications		Initial Procedure
	Ordinary Annuity	Annuity Due	
n, i, PMT, PV (Input any three and calculate the fourth.)	Direct reduction loan Discounted notes Mortgages	Leases	Use A or set BAL to zero.
n, i, PMT, PV, BAL (Input any four and calculate the fifth.)	Direct reduction loan with balloon Discounted notes with balloon	Leases with residual values	None
n, i, PMT, FV (Input any three and calculate the fourth.)	Sinking fund	Periodic savings insurance	Use A or set PV to zero.
n, i, PV, FV (Input any three and calculate the fourth.)	Compound amount Savings (Annuity mode is not applicable and has no effect)		Use A or set PMT to zero.

### Equations:

$$PV = \frac{PMT}{i} A [1 - (1 + i)^{-n}] + (BAL \text{ or } FV)(1 + i)^{-n}$$

where

$$A = \begin{cases} 1 & \text{ordinary annuity} \\ (1 + i) & \text{annuity due.} \end{cases}$$

### Remarks:

The equation above is solved for i using Newton's method where:

$$i_n = i_{n-1} - \frac{f(i_{n-1})}{f'(i_{n-1})}$$

This is why solutions involving PMT and i take longer than other solutions. It is quite possible to define problems which cannot be solved by this technique. Such problems usually result in an error message but may simply continue to run indefinitely.

Interest problems with balloon payments of opposite sign to the periodic payments may have more than one mathematically correct answer (or no answer at all). While this program may find one of the answers, it has no way of finding or indicating other possibilities.

A-000 LBLA		B-054 1	
A-001 CL X	Initialize by clearing PMT, PV, FV (BAL) registers	B-055 -	
A-002 S 3		B-056 X <sup>2</sup>	
A-003 S 4		B-057 R 3	
A-004 S 5		B-058 x	
A-005 JPBG	Jump to ordinary annuity	B-059 R 5	
B-000 LBLB	-----	B-060 -	-----
B-001 R .2		B-061 LBL7	-----
B-002 X=0		B-062 ÷	Guess for i.
B-003 GT00		B-063 CHS	-----
B-004 0	Annuity mode toggle 0 = annuity due	B-064 .	If guess is less than -0.9
B-005 S .2		B-065 9	use -0.9 for guess.
B-006 RTN		B-066 CHS	-----
B-007 LBL0		B-067 X≤Y	Store guess as a %.
B-008 1		B-068 X≥Y	-----
B-009 S .2		B-069 GS09	If guess = 0 stop.
B-010 RTN		B-070 X=0	-----
B-011 LBL1	-----	B-071 RTN	Calculate f(i)
B-012 0	Store dummy 0 for n.	B-072 LBL8	-----
B-013 S 1		B-073 GS00	-----
B-014 GS00	Calculate subroutine.	B-074 +	Calculate f'(i)
B-015 R 5	-----	B-075 R 4	-----
B-016 LSTX		B-076 +	-----
B-017 -		B-077 R 8	-----
B-018 R 4		B-078 R 1	-----
B-019 LSTX	Solve for n and store it in R <sub>1</sub> .	B-079 R 7	-----
B-020 +		B-080 ÷	-----
B-021 ÷		B-081 x	-----
B-022 CHS		B-082 S 6	-----
B-023 LN		B-083 R .0	-----
B-024 R 7		B-084 R 9	-----
B-025 LN		B-085 ÷	-----
B-026 ÷		B-086 -	-----
B-027 S 1		B-087 R .1	-----
B-028 RTN		B-088 x	-----
B-029 LBL2		B-089 R 0	-----
B-030 0	Clear R <sub>2</sub> for sum of i terms.	B-090 x	-----
B-031 S 2	-----	B-091 R 6	f(i)/f'(i)
B-032 R 5		B-092 R 5	Compute next i.
B-033 R 1		B-093 x	-----
B-034 R 3		B-094 -	Test increment to i for limit.
B-035 X=0	If PMT = 0, GTO n, i, PV, FV solution.	B-095 ÷	-----
B-036 GT06	-----	B-096 CHS	-----
B-037 x	Start guess of i.	B-097 GS09	-----
B-038 +	n PMT + BAL	B-098 R 2	-----
B-039 R 4	-----	B-099 ÷	-----
B-040 X=0	If PV = 0, GTO FV guess.	B-100 ABS	-----
B-041 GT09	-----	B-101 EEX	-----
B-042 +	PV guess for i.	B-102 CHS	-----
B-043 R 1		B-103 6	-----
B-044 ÷	n PMT + BAL + PV	B-104 X≤Y	Stop and display.
B-045 R 4	n	B-105 GT08	-----
B-046 GT07	-----	B-106 R 2	-----
B-047 LBL9		B-107 RTN	-----
B-048 R 5		B-108 LBL6	-----
B-049 LSTX		B-109 R 5	-----
B-050 +		B-110 R 4	-----
B-051 ENT†	FV guess for i.	B-111 ÷	-----
B-052 +		B-112 CHS	Compute i for n, i, PV, FV problem.
B-053 R 1		B-113 R 1	-----

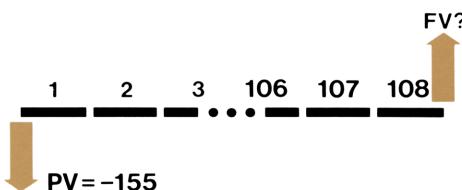
B-114 1/X		B-149 ÷	
B-115 YX		B-150 CHS	
B-116 1		B-151 S 5	
B-117 -	-----	B-152 RTN	-----
B-118 LBL9		B-153 LBL0	1 → R <sub>1</sub> for ordinary annuity.
B-119 EEX		B-154 1	
B-120 2	Convert i to % and add to R <sub>2</sub> .	B-155 S.1	-----
B-121 x		B-156 R 2	i → R <sub>9</sub> .
B-122 S+2		B-157 %	-----
B-123 RTN		B-158 S 9	1 + i → R <sub>7</sub> .
B-124 LBL3	-----	B-159 +	-----
B-125 1	Store dummy 1 for PMT.	B-160 S 7	-----
B-126 S 3	-----	B-161 R.2	If annuity due 1 + i → R <sub>1</sub> .
B-127 GSBO	Calculate subroutine.	B-162 X=0	
B-128 1/X	-----	B-163 X <sup>2</sup> Y	
B-129 R 4	-----	B-164 S.1	
B-130 RT↑		B-165 R 7	
B-131 +	Solve for PMT and store it in R <sub>3</sub> .	B-166 R 1	(1 + i) <sup>-n</sup> → R <sub>8</sub> .
B-132 x	-----	B-167 CHS	
B-133 CHS		B-168 YX	
B-134 S 3		B-169 S 8	
B-135 RTN	-----	B-170 R 5	
B-136 LBL4	-----	B-171 x	
B-137 1	Store dummy 1 for PV.	B-172 1	
B-138 S 4	-----	B-173 R 8	
B-139 GSBO	Calculate subroutine	B-174 -	1 - (1+i) <sup>-n</sup> → R <sub>0</sub>
B-140 +	-----	B-175 S.0	
B-141 CHS	Solve for PV and store it in R <sub>4</sub> .	B-176 R 3	PMT/i → R <sub>0</sub>
B-142 S 4	-----	B-177 R 9	
B-143 RTN		B-178 ÷	
B-144 LBL5	Calculate subroutine	B-179 S 0	
B-145 GSBO	-----	B-180 R.1	Calculate
B-146 R 4	Solve for FV(BAL) and store it in R <sub>5</sub> .	B-181 x	PMT
B-147 +		B-182 x	$\frac{1}{i} [1 - (1+i)^{-n}] R_1$
B-148 R 8		B-183 RTN	
<b>REGISTERS</b>			
0 PMT/i	1 n	2 i	3 PMT
4 PV	5 FV(BAL)	6 $n(1+i)^{-n-1}$	7 1+i
8 $(1+i)^{-n}$	9 $i/100$	.0 $1 - (1+i)^{-n}$	.1 1 or 1 + i
.2 Annuity flag	.3	.4	.5
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize. (1.00 = ordinary annuity)		A	1.00
3	If payments occur at the beginning of the period set annuity due mode*.		B	0.00/1.00

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
4	Input the known values:			
	Number of periods	n	STO [1]	n
	Periodic interest rate	i(%)	STO [2]	i (%)
	Periodic payment	PMT	STO [3]	PMT
	Present value	PV	STO [4]	PV
	Future value, balloon or balance	FV, (BAL)	STO [5]	FV, (BAL)
5	Calculate the unknown value.			
	Number of periods		GSB [1]	n
	Periodic interest rate		GSB [2]	i (%)
	Periodic payment		GSB [3]	PMT
	Present value		GSB [4]	PV
	Future value, balloon or balance		GSB [5]	FV, (BAL)
6	For a new case, go to step 4 and change appropriate values.  Input zero for any value not applicable in the new case.  *One or zero will be displayed alternately after pressing <b>B</b> , indicating that the ordinary annuity mode is on or off.			

**Example 1:**

You place \$155 in a savings account paying 5¾% compounded monthly. What sum of money may you withdraw at the end of 9 years?



**Keystrokes:**

A 155 CHS STO [4] →

**Outputs:**

-155.00

(Cash outlay)

5.75 ENTER ↴ 12 ÷ STO [2] →

0.48

(.48% monthly interest rate)

9 ENTER ↴ 12 × STO [1] →

108.00

(# months of compounding)

GSB [5] →

259.74

(FV)

If the interest is changed to 6%, what is the sum?

6 ENTER ↴ 12 ÷ STO [2] →

0.50

(.50% monthly interest rate)

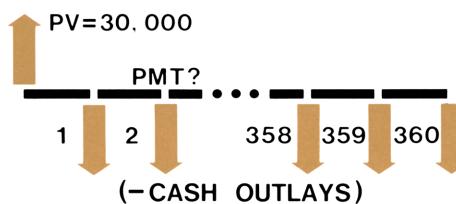
GSB [5] →

265.62

(FV)

**Example 2:**

You receive \$30000 from the bank as a 30 year, 9% mortgage. What monthly payment must you make to the bank to fully amortize the mortgage?

**Keystrokes:**

A 30 ENTER ↴ 12 × STO [1] →

**Outputs:**

360.00

(# monthly payments)

30000 STO [4] →

30000.00

(Loan amount)

9 ENTER ↴ 12 ÷ STO [2] →

0.75

(.75% monthly interest rate)

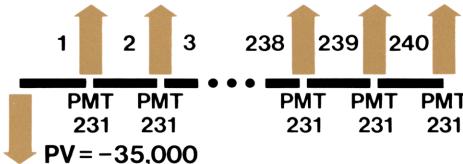
GSB [3] →

-241.39

(PMT)

**Example 3:**

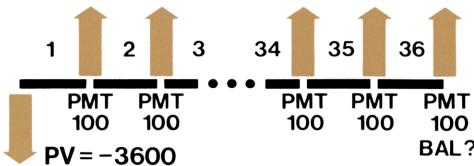
A fixed term annuity is available which requires a \$35,000 initial deposit. In return the depositor will receive monthly payments of \$231 for 20 years. What annual interest rate is being applied?

**Keystrokes:**

<b>Keystrokes:</b>	<b>Outputs:</b>	
A 35000 CHS STO 4	-35000.00	(Initial cash outlay)
231 STO 3	231.00	(Monthly income)
20 ENTER 12 x STO 1	240.00	(# monthly payments)
GSB 2	0.42	(0.42% monthly)
12 x	5.00	(5% annual interest rate)

**Example 4:**

Two individuals are constructing a loan with a balloon payment. The loan amount is \$3,600 and it is agreed that the annual interest rate will be 10% with 36 monthly payments of \$100. What balloon payment amount, to be paid coincident with the 36<sup>th</sup> payment, is required to fulfill the loan agreement? (Note the cashflow diagram below is with respect to the loaner. For the loanee, the appropriate diagram would be exactly the opposite.)

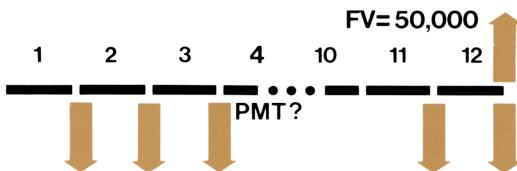
**Keystrokes:**

<b>Keystrokes:</b>	<b>Outputs:</b>
A 3600 CHS STO 4 10 ENTER 12 ÷ STO 2 36 STO 1 100 STO 3 GSB 5	675.27

(Note that the final payment is \$675.27 + \$100.00 = \$775.27 since the final payment falls at the end of the last period.)

**Example 5:**

A corporation has determined that a certain piece of equipment costing \$50,000 will be required in 3 years. Assuming a fund paying 7% compounded quarterly is available, what quarterly payment must be made in order to withdraw this cost from the fund if savings are to start at the end of this quarter?

**Keystrokes:**

```
A 50000 STO 5 3 ENTER 4 ×  
STO 1 7 ENTER  
4 ÷ STO 2 GSB 3
```

**Outputs:**

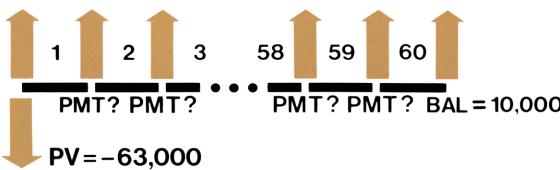
→ -3780.69

What single amount, invested immediately, would provide the same effect?

```
0 STO 3 GSB 4 → -40602.89
```

**Example 6:**

A “third party” leasing firm is considering the purchase of a mini-computer priced at \$63,000 and intends to achieve a 13% annual yield by leasing the computer to a customer for a 5-year period. Ownership is retained by the leasing firm, and at the end of the lease they expect to be able to sell the equipment for at least \$10,000. What should they establish as the monthly payments in order to realize their desired yield? (Since lease payments occur at the start of the periods, this is an annuity due problem.)

**Keystrokes:**

```
A B 63000 CHS STO 4 13 ENTER  
12 ÷ STO 2 5 ENTER 12 × STO 1  
10000 STO 5 GSB 3
```

**Outputs:**

→ 1300.16

If the price increased to \$70,000, what should the payments be?

70000 **CHS** **STO** **4** **GSB** **3** → 1457.73

If the payments were increased to \$1500 what would the yield be?

1500 **STO** **3** **GSB** **2** → 1.18 (% per month)

12 **x** → 14.12 (% per year)

### **Example 7:**

This program may also be used to calculate accumulated interest/remaining balance for loans. The accumulated interest between two points in time,  $n_1$  and  $n_2$ , is just the total payments made in that period less the principal reduction in that period. The principal reduction is the difference of the remaining balances for the two points in time. The following example demonstrates the concepts above.

For a 360 month, \$50000 loan at 9½% annual interest, find the remaining balance after the 24<sup>th</sup> payment and the accrued interest for payments 13–24 (between the 12<sup>th</sup> and 24<sup>th</sup> payments!).

First we must calculate the payment on the loan:

#### **Keystrokes:**

A 360 **STO** **1** 9.5 **ENTER**  
12 **÷** **STO** **2** 50000 **CHS** **STO**  
**4** **GSB** **3** →

#### **Outputs:**

420.43 payment

The remaining balance is found:

24 **STO** **1** **GSB** **5** → 49352.76 Remaining balance  
at month 24

Store this remaining balance and calculate the remaining balance at period 12:

**STO** **I** 12 **STO** **1** **GSB** **5** → 49691.68

The principal reduction between payments 12 and 24 is:

**I** **-** → 338.92

The accrued interest is 12 payments less the principal reduction:

**RCL** **3** 12 **x** → 5045.13 Total paid out  
**x<sub>2</sub>y** **-** → 4706.20 Accrued interest

## **NOTES**

## GAMES AND SUCH

### MOON ROCKET LANDER

Imagine for a moment the difficulties involved in landing a rocket on the moon with a strictly limited fuel supply. You're coming down tail-first, free-falling toward a hard rock surface. You'll have to ignite your rockets to slow your descent; but if you burn too much too soon, you'll run out of fuel 100 feet up, and then you'll have nothing to look forward to but cold eternal moon dust coming faster every second. The object, clearly, is to space your burns just right so that you will alight on the moon's surface with no downward velocity.

The game starts off with the rocket descending at a velocity of 50 feet/sec from a height of 500 feet. The velocity and height are shown in a combined display as -50.0500, the height appearing to the right of the decimal point and the velocity to the left, with a negative sign on the velocity to indicate downward motion. If a velocity is ever displayed with no fractional part, for example, -15., it means that you have crashed at a speed of 15 feet/sec. In game terms, this means that you have lost; in real-life, it signifies an even less favorable outcome.

You will start the game with 120 units of fuel. You may burn as much or as little of your available fuel as you wish at each step of your descent; burns of zero are quite common. A burn of 5 units will just cancel gravity and hold your speed constant. Any burn over 5 will act to change your speed in an upward direction. You must take care, however, not to burn more fuel than you have; for if you do, no burn at all will take place, and you will free-fall to your doom! The final velocity shown will be your impact velocity (generally rather high). You may display your remaining fuel at any time by recalling R<sub>7</sub>.

#### **Equations:**

We don't want to get too specific, because that would spoil the fun of the game; but rest assured that the program is solidly based on some old friends from Newtonian physics:

$$x = x_0 + v_0 t + \frac{1}{2} a t^2 \quad v = v_0 + a t \quad v^2 = v_0^2 + 2 a x$$

where x, v, a, and t are distance, velocity, acceleration, and time.

**Remarks:**

- If you crash before running out of fuel, the crash velocity shown will be the velocity before the burn, rather than the impact velocity.
- Use only integer values for burns. Any decimal entry will cause an error in the display for V.X.

A-000 LBL4 A-001 5 A-002 0 A-003 0 A-004 S 9 A-005 5 A-006 0 A-007 CHS A-008 S 8 A-009 1 A-010 2 A-011 0 A-012 S 7 A-013 LBL1 A-014 FIX4 A-015 R 9 A-016 EEX A-017 4 A-018 ÷ A-019 R 8 A-020 X<0 A-021 GT00 A-022 + A-023 GT02 A-024 LBL0 A-025 X#Y A-026 - A-027 LBL2 A-028 RTN A-029 R 7 A-030 X#Y A-031 X#Y A-032 GT03 A-033 S-7 A-034 5 A-035 - A-036 S 6 A-037 2 A-038 ÷ A-039 R 9 A-040 + A-041 R 8 A-042 + A-043 S 9 A-044 X<0 A-045 GT04 A-046 R 6 A-047 S+8 A-048 GT01 A-049 LBL3 A-050 R 8 A-051 X2 A-052 R 9 A-053 1 A-054 0 A-055 X A-056 + A-057 IX A-058 CHS A-059 S 8 A-060 LBL4 A-061 R 8 A-062 FIX0				
Initialize	If burn > fuel, prepare to crash.  Else update A, X, V.  $X \leftarrow X + V + A/2$ .			
Form display.	If X below ground, you've crashed.  Else update V: $V \leftarrow V + A$ .  All fuel gone, show crash velocity.			
V. X.	Crash V.			
<b>REGISTERS</b>				
0	1	2	3	
4	5	6	7	Fuel
8 V	9 X	.0	.1	
.2	.3	.4	.5	I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	-50.0500
3	Key in burn, display new			
	speed and altitude.	Burn	R S	V.X

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
4	Perform step 3 until you land or crash.			
5	To see remaining fuel at any time.		RCL [7]	Fuel
6	To display speed and altitude at any time.		GSB [1]	V.X
7	To start a new game, go to step 2.			

**Example:****Keystrokes:**

A	→	-50.0500	
0 [R/S]	→	-55.0448	
5 [R/S]	→	-55.0393	(Note constant V when burn = 5)
30 [R/S]	→	-30.0350	
0 [R/S]	→	-35.0318	
0 [R/S]	→	-40.0280	
0 [R/S]	→	-45.0238	
0 [R/S]	→	-50.0190	
RCL [7]	→	85.0000	(Remaining fuel)
GSB [1]	→	-50.0190	(Display V.X again)
10 [R/S]	→	-45.0143	
0 [R/S]	→	-50.0095	
RCL [7]	→	75.0000	
10 [R/S]	→	-45.0048	
25 [R/S]	→	-25.0013	
20 [R/S]	→	-25.	

Oops.

## SUBMARINE HUNT

Using your destroyer, you try to locate the position of the enemy submarine in a  $10 \times 10$  grid, and then to destroy it with your depth charge.

You input a seed (1 – 100) and the calculator will position the submarine in the center of one of the 100 squares (R, C), where R = row and C = column, and where R and C can each be 0, 1, 2, ..., 9.

You search for the submarine by taking sonar readings. If the submarine is in one of the 8 adjacent squares to your reading (or directly under your destroyer), the calculator will display “1.” Otherwise a “0” will be shown.

When you think you’ve located the submarine, move your destroyer directly over it (move to the same square) and drop a depth charge. A blinking “SOS” signal from the sub indicates a hit, while a “0” shows a miss. If you miss, the submarine will move randomly to one of the 4 adjacent squares in the same row or column.

You can make the hunt easier or more difficult. For an easier game, increase the sensitivity of your sonar, allowing you to detect the submarine as far away as 2 squares in any direction (you cover a square region of the ocean 5 squares on a side.) You can switch between 1- and 2-square sensitivity as often as you like during the game. You are charged two sonar readings, however, when you use the 2-square sensitivity.

To make a more challenging game, you can allow the submarine to move after each sonar echo as well as after each depth charge miss. The submarine always moves randomly to an adjacent square in the same row or column.

A depth charge has a range of 0.9. When you position your destroyer for a depth charge drop, you may move anywhere on the board. For instance, a depth charge dropped from a (2.5, 6.5) location would destroy any submarine in the center of squares (2, 6), (2, 7), (3, 6) and (3, 7).

Try to destroy the submarine using no more than 10 sonar readings and 1 depth charge, playing a regular game with 1 square sensitivity. You can check your status any time after sonar readings or depth charges.

Status format is dd.ss

Where dd = Number of depth charges fired  
ss = Number of sonar readings

Reference: This program is based on an HP-65 Users’ Library program written by Moshe Breiner.

A-000 LBLA		C-026 1	Select movement in row or column
A-001 CL R	Initialize	C-027 GT01	
A-002 .		C-028 LBL0	
A-003 5		C-029 2	
A-004 2		C-030 LBL1	
A-005 6		C-031 S I	
A-006 4		C-032 R i	
A-007 1		C-033 R 5	
A-008 6		C-034 +	
A-009 x		C-035 X<0	
A-010 S 0		C-036 GT00	
A-011 GSB7	Generate random submarine position.	C-037 9	Is move on playing board?
A-012 S 1		C-038 X>Y	
A-013 GSB7		C-039 X<Y	
A-014 S 2		C-040 GT01	
A-015 1		C-041 LBL0	
A-016 S.1		C-042 R 5	
A-017 0		C-043 2	
A-018 R/S		C-044 x	
A-019 LBL1		C-045 -	
A-020 R.0	Toggle for submarine movement after sonar.	C-046 LBL1	Store next position
A-021 1		C-047 S i	
A-022 X=Y		C-048 R 3	
A-023 -		C-049 RTN	
A-024 S.0		C-050 LBL8	
A-025 R/S		C-051 R 2	
A-026 LBL7		C-052 -	
A-027 JPC7		C-053 X>Y	
B-000 LBLB		C-054 R 1	
B-001 R.1		C-055 -	
B-002 1		C-056 +P	
B-003 X=Y		C-057 R 6	
B-004 +		C-058 -	
B-005 S.1		C-059 .	
C-000 LBLC	Sonar reading.	C-060 9	Response to sonar or depth charge.
C-001 R.1		C-061 -	
C-002 S+8		C-062 X<0	
C-003 S 6		C-063 GT00	
C-004 R↓		C-064 0	
C-005 GSB8		C-065 GT01	
C-006 R.0		C-066 LBL0	
C-007 S 5		C-067 1	
C-008 GSB9		C-068 LBL1	
C-009 R 3		C-069 S 3	
C-010 R/S		C-070 RTN	
C-011 JPD6		C-071 LBL7	
C-012 LBL9		C-072 FIX0	
C-013 GSB7		C-073 R 0	
C-014 4		C-074 9	
C-015 X>Y		C-075 9	
C-016 X>Y		C-076 7	
C-017 GT00		C-077 x	
C-018 R 5		C-078 FRAC	
C-019 CHS		C-079 S 0	
C-020 S 5		C-080 1	
C-021 LBL0		C-081 0	
C-022 GSB7		C-082 x	
C-023 5		C-083 INT	
C-024 X>Y	Movement of ±1 unit	C-084 RTN	
C-025 GT00		D-000 LBLD	

D-001 1	Depth charge.	D-020 5	
D-002 S+7		D-021 LBL1	
D-003 0		D-022 PSE	
D-004 S 6		D-023 DSZ	
D-005 R↓		D-024 GT01	
D-006 R↓		D-025 R/S	
D-007 GSB8		D-026 LBL6	
D-008 X#0	Compute response Hit?	D-027 FIX2	
D-009 GT00	-----	D-028 R 7	
D-010 1	-----	D-029 R 8	Display status
D-011 S 5	Move submarine after depth charge.	D-030 EEX	
D-012 GSB9	-----	D-031 ÷	
D-013 R/S	-----	D-032 +	
D-014 GT06	Submarine hit!	D-033 RTN	
D-015 LBL0		D-035 LBL8	Link to movement and response routines.
D-016 5		D-036 JPC8	
D-017 S I		D-037 LBL9	
D-018 5		D-038 JPC9	
D-019 0	Send "SOS"		

## REGISTERS

0	Seed	1 R	2 C	3 Response
4		5 ±1, 0	6 Used	7 dd
8	ss	9	.0 0,1 Movement after sonar.	.1 1,2 sonar sensitivity.
.2		.3	.4	.5 I Used

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in SEED (any number between 1 and 100).	SEED	A	0.
3	Select difficult game if desired (submarine always moving).		R.S	1.
4	To change sonar sensitivity: "2" means sensitivity dis- tance is 2 squares. "1" means sensitivity dis- tance is 1 square.		B	2. or 1.
5	Sonar "0" means no echo "1" means echo received or	R C	ENTER+ C	0. or 1.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	<b>Depth Charge</b>	R	<b>ENTER ↴</b>	
	"0" means miss	C	<b>D</b>	0. or
	Blinking "SOS" means HIT!			"SOS"
	To review status		<b>R/S</b>	dd.ss
5	For a new game, go to step 2.			

**Example 1:****Keystrokes:**58 **A** →**Outputs:**

0.

2 **ENTER ↴** 1 **C** →

1. Echo

You now know your enemy is in one of the "X" squares below.

	0	1	2	3	4	5	6	7	8	9	
9											9
8											8
7											7
6											6
5											5
4											4
3	x	x	x								3
2	x	*	x								2
1	x	x	x								1
0											0
	0	1	2	3	4	5	6	7	8	9	

Diagram of 1<sup>st</sup> move

The submarine cannot be in the X squares below.

1 **ENTER** 2 **C** → 0. No echo

	0	1	2	3	4	5	6	7	8	9	
9										9	
8										8	
7										7	
6										6	
5										5	
4										4	
3	x	x	x							3	
2	x	⊗	⊗							2	
1	x	⊗	⊗							1	
0										0	
	0	1	2	3	4	5	6	7	8	9	

Diagram of 2<sup>nd</sup> move

3 **ENTER** 0 **C** → 0. No echo

Check your status:

**R/S** → 0.03

You've narrowed down the submarine's location to just 2 squares, those containing an "X" with no circle.

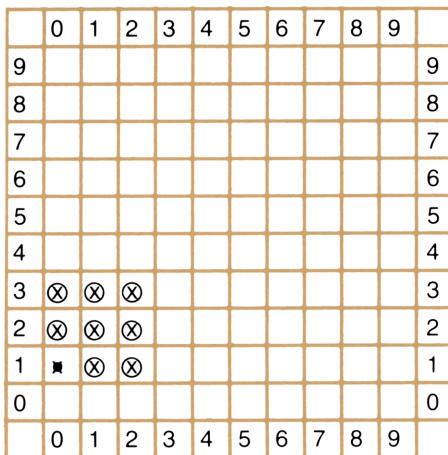
	0	1	2	3	4	5	6	7	8	9	
9										9	
8										8	
7										7	
6										6	
5										5	
4										4	
3	⊗	⊗	x							3	
2	⊗	⊗	⊗							2	
1	x	⊗	⊗							1	
0										0	
	0	1	2	3	4	5	6	7	8	9	

Diagram of 3<sup>rd</sup> move

1 **ENTER** 0 **C** →

1. Echo

This eliminates (3, 2) as a submarine location, so you've found it!

Diagram of 4<sup>th</sup> move1 **ENTER** 0 **D** →

“SOS”

A hit!

**R/S** →

1.04

Four sonar readings  
and one depth  
charge**Example 2:****Keystrokes:**60 **A** →**Outputs:**

0.

**R/S** →

1.

Submarine will now move on sonar echos as well as on depth charge misses.

7 **ENTER** 4 **C** →

1. Echo

The submarine is in one of the “X” squares in the left diagram below. But the submarine moves, so now it could be in any of the “X” squares in the right diagram below.

	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	
9											9			x	x	x					9
8		x	x	x							8		x	x	x	x	x				8
7		x	■	x							7		x	x	x	x	x				7
6		x	x	x							6		x	x	x	x	x				6
5											5			x	x	x					5
4											4										4
3											3										3
2											2										2
1											1										1
0											0										0
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	

Diagrams of 1<sup>st</sup> move8 **ENTER** 4 **C** →

0. No echo

You've eliminated some positions (left diagram below: X ), but new possible positions have been created by the enemy's random move (right diagram).

	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	
9			⊗	⊗	⊗						9		x			x					9
8		x	⊗	⊗	⊗	⊗	x				8	x	x	x	x	x	x	x	x	x	8
7		x	⊗	⊗	⊗	⊗	x				7	x	x	x	x	x	x	x	x	x	7
6		x	x	x	x	x					6	x	x	x	x	x	x	x	x	x	6
5		x	x	x							5	x	x	x	x	x					5
4											4		x	x	x						4
3											3										3
2											2										2
1											1										1
0											0										0
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	

Diagrams of 2<sup>nd</sup> move

7 ENTER 5 C →

1. Echo

This eliminates many possible positions (left diagram), but again, new ones are created (right diagram).

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9		⊗				⊗				9			x	x								9
8		⊗	⊗	⊗	x	x	⊗		8			x	x	x	x							8
7		⊗	⊗	⊗	x	■	x	⊗		7		x	x	x	x							7
6		⊗	⊗	⊗	x	x	x	⊗		6		x	x	x	x							6
5		⊗	⊗	⊗	⊗	⊗	⊗		5			x	x	x								5
4		⊗	⊗	⊗					4													4
3									3													3
2									2													2
1									1													1
0									0													0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

Diagrams of 3<sup>rd</sup> move

Fourth move. You try a depth charge.

7.5 ENTER 4.5 D → “SOS” A hit!  
 R/S → 1.03 dd.ss

The submarine used to be in one of these 4 squares:

	0	1	2	3	4	5	6	7	8	9													
9						x	x															9	
8					x	x	x	x	x													8	
7			x	x	x	x	x	x	x													7	
6			x	x	x	x	x	x	x													6	
5			x	x	x	x	x	x	x													5	
4																							4
3																							3
2																							2
1																							1
0																							0
	0	1	2	3	4	5	6	7	8	9													

RCL 1 → 8.00  
 RCL 2 → 4.00

This shows the submarine was sending its “SOS” from square (8,4).

	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9										9												9
8										8												8
7										7												7
6										6												6
5										5												5
4										4												4
3										3												3
2										2												2
1										1												1
0										0												0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9										9												9
8										8												8
7										7												7
6										6												6
5										5												5
4										4												4
3										3												3
2										2												2
1										1												1
0										0												0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	
9										9												9
8										8												8
7										7												7
6										6												6
5										5												5
4										4												4
3										3												3
2										2												2
1										1												1
0										0												0
	0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	

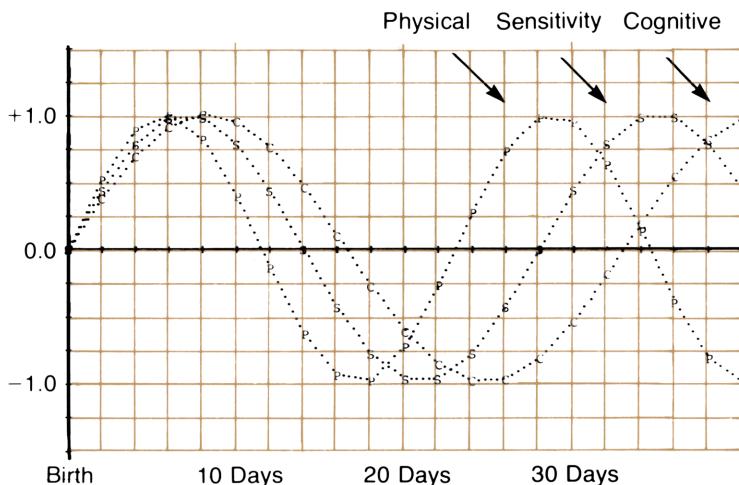
Playing boards for Submarine Hunt. You might wish to use copies of this page for your games.

## BIORHYTHMS

From the ancient days, philosophers and sages have taught that human happiness lies in the harmonious integration of body, mind, and heart. Now a twentieth-century theory claims to be able to quantitatively gauge the functioning of these three aspects of our selves: the physical, sensitive, and cognitive.

The biorhythm theory is based on the assumption that the human body has inner clocks or metabolic rhythms with constant cycle times. Currently, three cycles starting at birth in a positive direction are postulated. The 23-day or physical cycle relates with physical vitality, endurance and energy. The 28-day or sensitivity cycle relates with sensitivity, intuition and cheerfulness. The 33-day or cognitive cycle relates with mental alertness and judgement.

For each cycle, a day is considered either high, low, or critical. Let  $x$  be the output value for a given cycle. The high ( $0 < x \leq 1$ ) times are regarded as energetic times, you are your most dynamic in the cycle. The low ( $-1 \leq x < 0$ ) times are regarded as the recuperative periods. The critical days ( $x = 0$ ) are regarded as your accident prone days especially for the physical and sensitivity cycles.



**Remarks:**

1. The birthdate and biodate must be between January 1, 1901, and December 31, 2099.
2. The format for input of dates is MM.DDYYYY. For example, June 3, 1976, is keyed in as 6.031976. The program does not check input data. Thus, if an improper format or an invalid date (e.g., February 30) is keyed in, erroneous answers may result.
3. This program sets the angular mode to radians (RAD).

A-000 LBLA		A-044 1	Next day
A-001 RAD	Birthdate	A-045 S+8	-----
A-002 GSB9	Store N	A-046 GSB8	
A-003 S 9	-----	A-047 GT06	
A-004 RTN		A-048 LBL9	
A-005 LBL1	Biodate	A-049 ENT1	Compute N(M, D, Y)
A-006 GSB9	-----	A-050 INT	
A-007 R 9		A-051 S 6	M
A-008 -		A-052 -	
A-009 S 8	Store N <sub>2</sub> - N <sub>1</sub> .	A-053 EEX	
A-010 LBL8		A-054 2	
A-011 1		A-055 x	
A-012 8		A-056 ENT↑	D
A-013 S 7	23-day cycle	A-057 INT	
A-014 SPC	28-day cycle	A-058 S 5	
A-015 GSB7	-----	A-059 -	
A-016 GSB7		A-060 EEX	
A-017 LBL7	-----	A-061 4	
A-018 5	# days	A-062 x	Y
A-019 S+7		A-063 S 4	
A-020 R 8		A-064 2	
A-021 R ?		A-065 R 6	
A-022 ÷		A-066 XY	
A-023 FRAC		A-067 GT02	
A-024 2		A-068 1	
A-025 x		A-069 S-4	
A-026 #		A-070 1	
A-027 x		A-071 2	
A-028 SIM		A-072 S+6	
A-029 ENT1		A-073 LBL2	
A-030 ABS		A-074 1	
A-031 X#0		A-075 S+6	
A-032 ÷		A-076 R 6	
A-033 LSTX		A-077 3	
A-034 EEX		A-078 0	
A-035 7		A-079 .	
A-036 +		A-080 6	
A-037 EEX		A-081 x	
A-038 7		A-082 INT	
A-039 -		A-083 R 4	
A-040 x		A-084 3	
A-041 PRTX	Bio value	A-085 6	
A-042 RTN	-----	A-086 5	
A-043 LBL6		A-087 .	

A-088	Z		A-092	+	
A-089	5		A-093	R 5	
A-090	X		A-094	+	
A-091	INT				
<b>REGISTERS</b>					
0	1	2	3		
4 Y	5 D	6 M	7	23, 28, 33	
8 N <sub>2</sub> - N <sub>1</sub>	9 N <sub>1</sub>	.0	.1		
.2	.3	.4	.5	I	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in birthdate.	MM.DDYYYY	A	day#*
3	Key in biodate and find bio values.	MM.DDYYYY	GSB 1	P
				S
				C
4	To find bio values for succeeding days.		R,S	P
				S
				C
5	For a new birthdate, go to step 2; for a new biodate, go to step 3.			
	*See Calendar Functions for an explanation of this number.			

**Example:**

Calculate the Bio values for June 29, 1976, for a person born March 27, 1948. Find the values for the two days following also.

**Keystrokes:**

3.271948 A

**Outputs:**

6.291976 GSB 1	→	-1.00 *** (P) -0.62 *** (S) -1.00 *** (C)
R/S	→	-0.98 *** (June 30) -0.78 *** -0.97 ***
		-0.89 *** (July 1) -0.90 *** -0.91 ***

R/S halts output.

## COUNT-DOWN TIMER

This program provides a count-down timer and a calibration routine for measuring elapsed time. When using this program, you should remember that the clock circuits of the HP-95C are designed for calculator use, not for accurate time keeping. Although the routine may be calibrated quite accurately, highly stable performance should not be expected due to variable conditions about the calculator.

### Equations:

$$C_{a_{\text{new}}} = C_{a_{\text{old}}} \frac{\text{HP-95C time}}{\text{actual time}}$$

A-000 LBLA A-001 FIX4 A-002 S 0 A-003LBL2 A-004 0 A-005 PRTX A-006 R/S A-007 S 1 A-008 +H A-009 R 0 A-010 X A-011 S I A-012 R 1 A-013 R/S A-014LBL1	Store constant. ----- "Alarm" ----- Store time. ----- Store counter. -----	A-015 DSZ H-016 GT01 A-017 GT02 B-000LBL6 B-001HMS- B-002 +H B-003 R 1 B-004 +H B-005 ÷ B-006 1/X B-007 R 0 B-008 X B-009 PRTX B-010 GT04	Loop on counter. ----- Go to "alarm". ----- Calibrate constant. -----
<b>REGISTERS</b>			
0 Ca	1 Time	2	3
4	5	6	7

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in timer constant (try 17000)	C <sub>a</sub>	A	0.0000
3	Key in desired time.	t H.MS	R/S	t
4	Start timer.		R/S	0.0000
5	Timer loops for time t. When 0.0000 is displayed, time has elapsed. For new time t, execute steps 3 and 4. To cali- brate, proceed to step 6.			
6	Key in ending time and starting time and compute new constant.	t <sub>e</sub>  t <sub>s</sub>	ENTER ↴  B	C <sub>a</sub>
	For new time t, execute steps 3 and 4.			
	The program executes a PRINTx to provide an audible signal for its completion of the timing. This may be omitted from the program (A-005) if desired.			

**Example:**

Measure elapsed times of 35 seconds and 1 minute 8 seconds.

First, key in the timer constant:

**Keystrokes:**

17000 A →  
0.0035 R/S →

**Outputs:**

0.0000  
0.0035

Start timer:

R/S → 0.0000 \*\*\*

Timer runs for approximately 35 seconds.

For second desired time:

0.0108 R/S → 0.0108

Start timer:

R/S → 0.0000 \*\*\*

Supposing you had noticed the *actual* ending and starting times of the second example were 9:58:16 and 9:57:01, respectively. Calibrate the timer with this information:

9.5816 ENTER 9.5701 B → 15413.3336

Now try the calibrated timer for 2 minutes 5 seconds:

0.0205 R/S → 0.0205

Start timer:

R/S → 0.0000 \*\*\*

Under the same conditions, the new timer constant 15413 should be keyed in to A for subsequent use of this program. Your HP-95C will have its own "best" constant for calibration.

## CALENDAR FUNCTIONS

For the period March 1, 1900 through February 28, 2100, these programs solve for dates and days.

Given a date, the first program computes an associated day number\*. By using this program on two dates, the number of days between those dates may be found.

The second program takes a day number\* and finds the corresponding date.

The third program computes the day of the week from a given day number\*.

By using the first two programs together, a second date may be computed from a date and a specified number of days (see example).

A date must be input in mm.ddyyyy format. For instance, June 3, 1975, is keyed in as 06.031975. It is important that the zero between the decimal point and the day of the month be included when the day of the month is less than 10. The day of the week is represented by the digits 0 through 6 where zero is Sunday.

\*The Julian Day number is an astronomical convention representing the number of days since January 1, 4713 B.C.

**Equations:**

To compute the day number from the date:

$$\text{Julian Day number} = \text{INT}(365.25 y') + \text{INT}(30.6001 m') + d + 1,720,982$$

where

$$y' = \begin{cases} \text{year} - 1 & \text{if } m = 1 \text{ or } 2 \\ \text{year} & \text{if } m > 2 \end{cases}$$

$$m' = \begin{cases} \text{month} + 13 & \text{if } m = 1 \text{ or } 2 \\ \text{month} + 1 & \text{if } m > 2 \end{cases}$$

Then days between dates is found by

$$\text{Days} = \text{Day number}_2 - \text{Day number}_1$$

To compute the date from a day number:

$$\text{Day \#} = \text{Julian Day Number} - 1,720,982$$

$$y' = \text{INT} \left[ \frac{\text{Day \#} - 122.1}{365.25} \right]$$

$$m' = \text{INT} \left[ \frac{\text{Day \#} - \text{INT}(365.25 y')}{30.6001} \right]$$

$$\begin{aligned} \text{Day of the month} &= \text{Day \#} - \text{INT}[365.25 y'] \\ &\quad - \text{INT}[30.6001 m'] \end{aligned}$$

$$\text{Month} = m = \begin{cases} m' - 13 & \text{if } m' = 14 \text{ or } 15 \\ m' - 1 & \text{if } m' < 14 \end{cases}$$

$$\text{Year} = \begin{cases} y' & \text{if } m > 2 \\ y' + 1 & \text{if } m = 1 \text{ or } 2 \end{cases}$$

To compute the day of the week:

$$\text{Day of the week} = 7 \times \text{FRAC}[(\text{Day \#} + 5)/7]$$

**Remarks:**

No checking is done to determine if input data represents valid dates.

Although the programs have been written to function independently, some programming savings can be made by the user if the programs are used concurrently.

Storage registers R<sub>0</sub>–R<sub>3</sub> are available to the user.

A-000 LBLA		B-003 2	
A-001 ENT1	Break date input into the individual components of mm,dd,yyyy.	B-004 2	
A-002 INT		B-005 .	
A-003 S 7		B-006 1	
A-004 -		B-007 -	
A-005 EEX		B-008 3	Calculate y'
A-006 2		B-009 6	
A-007 X		B-010 5	
A-008 ENT1		B-011 .	
A-009 INT		B-012 2	
A-010 S 8		B-013 5	
A-011 -		B-014 S 5	
A-012 EEX		B-015 .	
A-013 4		B-016 INT	
A-014 X		B-017 S 9	
A-015 S 9		B-018 R 5	
A-016 R 7		B-019 X	
A-017 1	m + 1	B-020 INT	
A-018 +		B-021 R 4	
A-019 ENT1		B-022 -	
A-020 1/X		B-023 CHS	
A-021 *	m + 1 → m' y → y'	B-024 S 4	
A-022 7		B-025 3	
A-023 +		B-026 0	Calculate m'
A-024 CHS		B-027 .	
A-025 INT		B-028 6	
A-026 S+9	If input to this routine has absolute value 1 or greater: y = y ± 1 (+ for plus input)	B-029 0	
A-027 1		B-030 0	
A-028 2		B-031 1	
A-029 X		B-032 S 6	
A-030 -		B-033 .	
A-031 3		B-034 INT	
A-032 0		B-035 S 7	
A-033 .		B-036 R 4	
A-034 6		B-037 X*Y	
A-035 0		B-038 R 6	Calculate day of month.
A-036 0		B-039 X	
A-037 1		B-040 INT	
A-038 S 6		B-041 -	
A-039 X	Compute day number.	B-042 S 8	
A-040 INT		B-043 R 7	
A-041 R 9		B-044 1	
A-042 3		B-045 R 8	
A-043 6		B-046 %	
A-044 5		B-047 -	
A-045 .		B-048 -	
A-046 2		B-049 R 7	
A-047 5		B-050 1	
A-048 S 5		B-051 4	
A-049 X		B-052 .	
A-050 INT		B-053 INT	Correct m' - 1 and y' to m and y.
A-051 +		B-054 S+9	
A-052 R 8		B-055 1	
A-053 +		B-056 2	
A-054 FIX0		B-057 X	
A-055 RTN		B-058 -	
B-000 LBLB		B-059 R 9	
B-001 S 4		B-060 EEX	
B-002 1		B-061 6	Finish building mm.ddyyyy result and display answer.

B-062 ÷ B-063 + B-064 FIX6 B-065 RTN C-000 LBLC C-001 5 C-002 +		C-003 7 C-004 ÷ C-005 FRAC C-006 7 C-007 × C-008 RTN	Calculate day of the week from day #.
<b>REGISTERS</b>			
0	1	2	3
4 Day #	5 365.25	6 30.6001	7 m
8 d	9 y	.0	.1
.2	.3	.4	.5
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	For day #'s from dates.			
1	Key in program A.			
2	Key in date and compute day #.	date	A	day #
3	Repeat step 2 for any desired date from March 1, 1900 to February 28, 2100.			
4	For # days between dates, compute day #'s for each and find difference.	date <sub>1</sub>	A STO [1]	day #1 day #1
		date <sub>2</sub>	A RCL [1]	day #2 day #1
			-	day #2 - day #1
	For dates from day #'s			
1	Key in program B.			
2	Key in day # and compute date.	day #	B	date
	For day of the week from day #.			
1	Key in program C.			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
2	Key in day # and compute day of the week.			
		day #	C	0, ..., 6

**Example 1:**

Senior Lieutenant Yuri Gagarin flew Vostok I into space on April 12, 1961. On July 21, 1969, Neil Armstrong set foot on the moon. How many days had passed between the first manned space flight and the moon landing? On what day of the week did each event take place.

**Keystrokes:**

4.121961 A STO 1 →  
 7.211969 A STO 2 →  
 RCL 1 - →  
 RCL 1 C →  
 RCL 2 C →

**Outputs:**

716420. (Day #1)  
 719442. (Day #2)  
 3022. (Days)  
 3. (Wednesday)  
 1. (Monday)

**Example 2:**

A short term note is due in 200 days. If the issue date is June 11, 1976, what is the maturity date?\*

**Keystrokes:**

6.111976 A →  
 200 + →  
 B →

**Outputs:**

721959.  
 722159.  
 12.281976 (December 28,  
 1976)

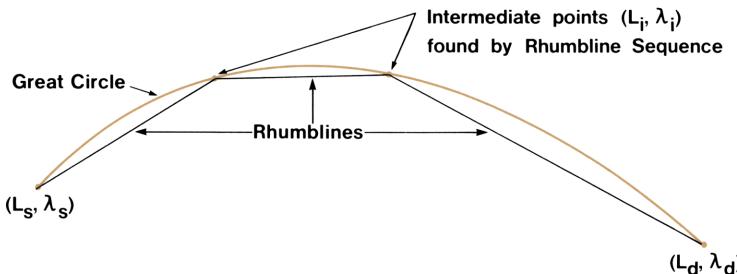
\*Note this calculation requires both the A and B programs be entered in the calculator. First a day number is computed for the known date, the number of days (200) is added to it, and this new day number is converted to a date.

Some securities use a 30/360 day calendar while this program performs all calculations using the actual number of days. Do not use the program for financial purposes unless you are sure that actual calendar days are correct.

# NAVIGATION

## RHUMBLINE SEQUENCE

This program is designed to assist in the activity of course planning. You must supply the source and destination latitude and longitude. Then the rhumbline course and distance from source to destination may be found.



If a longitude interval ( $\Delta\lambda$ ) is then keyed in, the program will compute a sequence of rhumbines which approximate the great circle route from source to destination, where the rhumbines join points separated in longitude by  $\Delta\lambda$ . The output will be as follows: latitude (L) and longitude ( $\lambda$ ) of source, course (C) and distance (D) from source to first intermediate point, L and  $\lambda$  of first intermediate point, C and D from first to second intermediate point, L and  $\lambda$  of second intermediate point, and so on, the final outputs being L and  $\lambda$  of the destination. At the end of all calculations, the sum of all the rhumbine distances is printed out.

The next intermediate point is always found on the great circle from the previous point to the destination. Thus, the points do not all lie on the same great circle from source to destination.

The longitude interval  $\Delta\lambda$  should be keyed in as a positive number if the course is westward or as a negative number if the course is eastward.

Latitudes and longitudes are input and output in degrees-minutes-seconds format; course and  $\Delta\lambda$  are in decimal degrees. Southern latitudes and eastern longitudes are input and output as negative numbers.

### Equations:

$$L_i = \tan^{-1} \left[ \frac{\tan(L_d) \sin(\lambda_i - \lambda_{i-1}) - \tan(L_{i-1}) \sin(\lambda_i - \lambda_d)}{\sin(\lambda_d - \lambda_{i-1})} \right]$$

$$C = \tan^{-1} \frac{\pi(\lambda_{i-1} - \lambda_i)}{180 [\ln \tan(45 + \frac{1}{2}L_i) - \ln \tan(45 + \frac{1}{2}L_{i-1})]}$$

$$D = \begin{cases} 60(\lambda_i - \lambda_{i-1})\cos L_i; & \cos C = 0 \\ 60 \frac{(L_i - L_{i-1})}{\cos C}; & \text{otherwise} \end{cases}$$

where

$L_0, \lambda_0 = L_s, \lambda_s$  = latitude, longitude of source

$L_d, \lambda_d$  = latitude, longitude of destination

$L_i, \lambda_i$  = latitude, longitude of  $i^{\text{th}}$  point.

### Remarks:

This program assumes the calculator is set in DEG mode.

$A-000 LBLA$ $A-001 \rightarrow H$ $A-002 S 0$ $A-003 S 6$ $A-004 RTN$ $A-005 \rightarrow H$ $A-006 S 1$ $A-007 S 7$ $A-008 RTN$ $A-009 \rightarrow H$ $A-010 S 2$ $A-011 S 8$ $A-012 RTN$ $A-013 \rightarrow H$ $A-014 S 3$ $A-015 S 9$ $A-016 RTN$ $A-017 JPB9$ $B-000 LBLB$ $B-001 FIX4$ $B-002 S 4$ $B-003 R 1$ $B-004 R 3$ $B-005 -$ $B-006 2$ $B-007 \div$ $B-008 SIN$ $B-009 SIN^{-1}$ $B-010 \hat{z}$ $B-011 x$ $B-012 R 4$ $B-013 \div$ $B-014 ABS$ $B-015 EEX$ $B-016 CHS$ $B-017 7$ $B-018 -$ $B-019 INT$	$L_1$ $\lambda_1$ $L_2$ $\lambda_2$ Compute rhumbline C, D. $\Delta\lambda$	$B-020 S I$ $B-021 0$ $B-022 S 5$ $B-023 SPC$ $B-024 R 0$ $B-025 S 6$ $B-026 \rightarrow HMS$ $B-027 PRTX$ $B-028 R 1$ $B-029 S 7$ $B-030 \rightarrow HMS$ $B-031 PRTX$ $B-032 SPC$ $B-033 R 1$ $B-034LBL1$ $B-035 R 4$ $B-036 +$ $B-037 1$ $B-038 \rightarrow R$ $B-039 \rightarrow P$ $B-040 R \downarrow$ $B-041 S 9$ $B-042 GS87$ $B-043 S 8$ $B-044 GS89$ $B-045 S+5$ $B-046 SPC$ $B-047 R 8$ $B-048 S 6$ $B-049 \rightarrow HMS$ $B-050 PRTX$ $B-051 R 9$ $B-052 S 7$ $B-053 \rightarrow HMS$ $B-054 PRTX$ $B-055 SPC$ $B-056 R 9$ $B-057 DSZ$	Clear $R_s$ for $\Sigma$ .  Print $L_1, \lambda_1$ .  Loop to compute successive rhumbines.  $\lambda_i$ $L_i$ Rhumb line from $(L_{i-1}, \lambda_{i-1})$ to $(L_i, \lambda_i)$  $L_i \rightarrow L_{i-1}$ $\lambda_i \rightarrow \lambda_{i-1}$
---	--	---	---

<b>B-058 GT01</b>		<b>B-111</b> $\div$	
<b>B-059 R 2</b>	$L_i \leftarrow L_2$	<b>B-112</b> $\neq$	
<b>B-060 S 8</b>	$\lambda_i \leftarrow \lambda_2$	<b>B-113</b> $\times$	
<b>B-061 R 3</b>		<b>B-114 R 8</b>	
<b>B-062 S 9</b>	$\Sigma D$	<b>B-115 GSB4</b>	
<b>B-063 GSB9</b>		<b>B-116 R 6</b>	
<b>B-064 S+5</b>		<b>B-117 GSB4</b>	
<b>B-065 R 2</b>	$L_2$	<b>B-118</b> $-$	
<b>B-066 +HMS</b>		<b>B-119 +P</b>	
<b>B-067 PRTX</b>		<b>B-120 R<math>\downarrow</math></b>	
<b>B-068 R 3</b>	$\lambda_2$	<b>B-121 S.1</b>	
<b>B-069 +HMS</b>		<b>B-122 R.0</b>	
<b>B-070 PRTX</b>		<b>B-123 SIN</b>	
<b>B-071 SPC</b>	Final $\Sigma D$	<b>B-124 SIN<math>^{-1}</math></b>	$x < 0$ means East to West. W to E, C is answer.
<b>B-072 SPC</b>		<b>B-125 X<math>\downarrow</math>D</b>	
<b>B-073 R 5</b>		<b>B-126 GT00</b>	
<b>B-074 FIX1</b>		<b>B-127 R.1</b>	
<b>B-075 PRTX</b>		<b>B-128 GT02</b>	
<b>B-076 RTN</b>		<b>B-129 LBL0</b>	
<b>B-077 LBL7</b>	$\lambda_i \rightarrow L_i$	<b>B-130</b> 3	E to W, 360 - C.
<b>B-078 R 7</b>	Computes $L_i$ for $\lambda_i$ on great circle from $(L_{i-1}, \lambda_{i-1})$ to $(L_2, \lambda_2)$ .	<b>B-131</b> 6	
<b>B-079</b> -		<b>B-132</b> 0	
<b>B-080 SIN</b>		<b>B-133 R.1</b>	
<b>B-081 R 2</b>		<b>B-134 ABS</b>	
<b>B-082 TAN</b>		<b>B-135</b> -	
<b>B-083</b> $\times$		<b>B-136 LBL2</b>	
<b>B-084 R 9</b>		<b>B-137 ABS</b>	
<b>B-085 R 3</b>		<b>B-138 PRTX</b>	
<b>B-086</b> -		<b>B-139 R.0</b>	
<b>B-087 SIN</b>		<b>B-140 R 8</b>	
<b>B-088 R 6</b>		<b>B-141 COS</b>	
<b>B-089 TAN</b>		<b>B-142</b> $\times$	
<b>B-090</b> $\times$		<b>B-143 ENT<math>\uparrow</math></b>	
<b>B-091</b> -		<b>B-144 R 8</b>	
<b>B-092 R 3</b>		<b>B-145 R 6</b>	
<b>B-093 R 7</b>		<b>B-146</b> -	
<b>B-094</b> -		<b>B-147 R.1</b>	
<b>B-095 SIN</b>		<b>B-148 COS</b>	
<b>B-096</b> $\div$		<b>B-149 X<math>\neq</math>0</b>	
<b>B-097 TAN<math>\downarrow</math></b>		<b>B-150</b> $\div$	
<b>B-098 RTN</b>		<b>B-151 ENT<math>\uparrow</math></b>	
<b>B-099 LBL9</b>	Rhumb line C, D from $(L_{i-1}, \lambda_{i-1})$ to $(L_i, \lambda_i)$ .	<b>B-152 X=0</b>	
<b>B-100 FIX1</b>		<b>B-153 R<math>\uparrow</math></b>	
<b>B-101 R 7</b>		<b>B-154</b> 6	
<b>B-102 R 9</b>		<b>B-155</b> 0	
<b>B-103</b> -		<b>B-156</b> $\times$	
<b>B-104 S.0</b>		<b>B-157 ABS</b>	
<b>B-105</b> 2		<b>B-158 PRTX</b>	
<b>B-106</b> $\div$		<b>B-159 SPC</b>	
<b>B-107 SIN</b>	Make $-180 \leq \lambda_{i-1} - \lambda_i \leq 180$ .	<b>B-160 FIX4</b>	
<b>B-108 SIN<math>\downarrow</math></b>		<b>B-161 RTN</b>	
<b>B-109</b> 9		<b>B-162 LBL4</b>	
<b>B-110</b> 0		<b>B-163</b> 2	
			Distance.

<b>B-164</b>	<b>÷</b>			<b>B-167</b>	<b>+</b>	
<b>B-165</b>	<b>4</b>			<b>B-168 TAN</b>		
<b>B-166</b>	<b>5</b>			<b>B-169 LN</b>		
<b>REGISTERS</b>						
0	$L_i$	1 $\lambda_i$	2 $L_2$	3 $\lambda_2$		
4	$\Delta\lambda$	5 $\Sigma D$	6 $L_{i-1}$	7 $\lambda_{i-1}$		
8	$L_i$	9 $\lambda_i$	.0 $\lambda_i - \lambda_{i-1}$	.1 Used		
.2		.3	.4	.5	I	Used

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in latitude and longitude of source.	$L_s$	A	
		$\lambda_s$	R/S	
3	Key in latitude and longitude of destination.	$L_d$	R/S	
		$\lambda_d$	R/S	
4	(optional) Compute rhumbline course and distance from source to destination.		R/S	C (dec. deg.) D (n. m.)
5	Key in longitude interval and compute rhumbline sequence.	$\Delta\lambda$	B	$L_i$ $\lambda_i$ C D $L_i$ $\lambda_i$ C

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
				D
				$L_d$
				$\lambda_d$

**Example:**

Find the intended track for a voyage from Tokyo to San Francisco which follows rhumbines approximating the great circle track. Use intermediate points separated by  $20^\circ$  in longitude. Since the voyage is eastward, use  $\Delta\lambda = -20$ . Tokyo is at ( $L35^{\circ}40'N$ ,  $\lambda139^{\circ}45'E$ ), San Francisco at ( $L37^{\circ}49'N$ ,  $\lambda122^{\circ}25'W$ ).

**Keystrokes:**

35.40 A 139.45 CHS R/S →

37.49 R/S 122.25 R/S →

20 CHS B →

**Outputs:**

-139.75

122.42

35.4000 \*\*\* ( $L_s$ )-139.4500 \*\*\* ( $\lambda_s$ )

60.5 \*\*\* (C)

1053.7 \*\*\* (D)

44.1829 \*\*\* ( $L_1$ )-159.4500 \*\*\* ( $\lambda_1$ )

74.4 \*\*\* (C)

861.1 \*\*\* (D)

48.0934 \*\*\* ( $L_2$ )-179.4500 \*\*\* ( $\lambda_2$ )

89.3 \*\*\* (C)

799.3 \*\*\* (D)

48.1843 \*\*\* ( $L_3$ )160.1500 \*\*\* ( $\lambda_3$ )

104.3 \*\*\* (C)

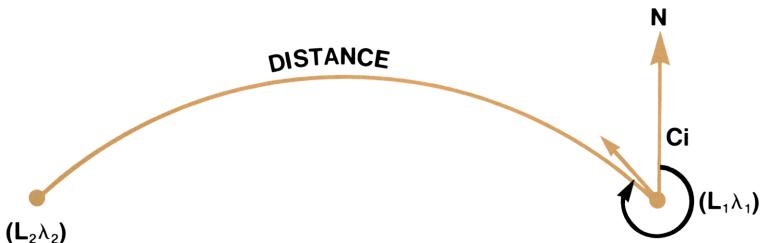
851.0 \*\*\* (D)

44.4839 \*\*\* ( $L_4$ )140.1500 \*\*\* ( $\lambda_4$ )

117.6 \*\*\* (C)  
 905.5 \*\*\* (D)  
 37.4900 \*\*\* ( $L_d$ )  
 122.2500 \*\*\* ( $\lambda_d$ )  
 4470.7 \*\*\* (Total D)

## GREAT CIRCLE NAVIGATION

This program computes the great circle distance between two points and the initial course from the first point. Coordinates are input in degrees-minutes-seconds format. Outputs are distance in nautical miles and initial course in decimal degrees.



### Equations:

$$D = 60 \cos^{-1} [\sin L_1 \sin L_2 + \cos L_1 \cos L_2 \cos (\lambda_2 - \lambda_1)]$$

$$C = \cos^{-1} \left[ \frac{\sin L_2 - \sin L_1 \cos (D/60)}{\sin (D/60) \cos L_1} \right]$$

$$C_i = \begin{cases} C & ; \sin (\lambda_2 - \lambda_1) < 0 \\ 360 - C; & \sin (\lambda_2 - \lambda_1) \geq 0 \end{cases}$$

where

$L_1, \lambda_1$  = coordinates of initial point

$L_2, \lambda_2$  = coordinates of final point

D = distance from initial to final point

$C_i$  = initial course from initial to final point

### Remarks:

Southern latitudes and eastern longitudes must be entered as negative numbers. Truncation and round off errors occur when the source and destination are very close together (1 mile or less).

Do not use coordinates located at diametrically opposite sides of the earth.

Do not use latitudes of  $+90^{\circ}$  or  $-90^{\circ}$ .

Do not try to compute initial heading along a line of longitude ( $L_1 = L_2$ ).

This program assumes the calculator is set in DEG mode.

A-000 LBLA A-001 →H A-002 S 0 A-003 RTN A-004 →H A-005 S 1 A-006 R/S A-007 →H A-008 S 2 A-009 R/S A-010 →H A-011 S 3 A-012 R/S B-000LBLB B-001 R 0 B-002 SIN B-003 R 2 B-004 SIN B-005 x B-006 R 0 B-007 COS B-008 R 2 B-009 COS B-010 x B-011 R 3 B-012 R 1 B-013 - B-014 S 4 B-015 COS B-016 x B-017 + B-018 S 5 B-019 COS⁻¹		L <sub>1</sub> ----- $\lambda_1$ ----- L <sub>2</sub> ----- $\lambda_2$ -----	B-020 S 6 B-021 6 B-022 0 B-023 x B-024 PRTX B-025 R 2 B-026 SIN B-027 R 0 B-028 SIN B-029 R 5 B-030 x B-031 - B-032 R 0 B-033 COS B-034 ÷ B-035 R 6 B-036 SIN B-037 ÷ B-038 COS⁻¹ B-039 R 4 B-040 SIN B-041 X $\times$ 0 B-042 GT00 B-043 R↓ B-044 3 B-045 6 B-046 0 B-047 X $\div$ Y B-048 - B-049 RTN B-050LBLD B-051 R↓ B-052 PRTX	D. C. C <sub>i</sub> C <sub>i</sub>
<b>REGISTERS</b>				
0	L <sub>1</sub>	1	$\lambda_1$	2
4	$\lambda_2 - \lambda_1$	5	cos D/60	6
8		9		.0
.2		.3		.4
				.5
				1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in latitude and longitude			
	of source.	L <sub>1</sub> (D.MS)	[A]	L <sub>1</sub> (dec. deg.)

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
		$\lambda_1$ (D.MS)	[R/S]	$\lambda_1$ (dec. deg.)
3	Key in latitude and longitude of destination.	$L_2$ (D.MS)	[R/S]	$L_2$ (dec. deg.)
		$\lambda_2$ (D.MS)	[R/S]	$\lambda_2$ (dec. deg.)
4	Compute distance and initial course.		[B]	D (n.m.)
				$C_i$ (dec. deg.)

**Example:**

Find the distance and initial course for the great circle from Tokyo (L $35^{\circ}40'N$ ,  $\lambda 139^{\circ}45'E$ ) to San Francisco (L $37^{\circ}49'N$ ,  $\lambda 122^{\circ}25'E$ ).

**Keystrokes:**

35.40 [A] 139.45 [CHS] [R/S] →

37.49 [R/S] 122.25 [R/S] →

[B] →

**Outputs:**

-139.75

122.42

4460.04 \*\*\* (D)  
54.37 \*\*\* ( $C_i$ )**SIGHT REDUCTION TABLE**

This program calculates the computed altitude  $H_c$  and azimuth  $Z_n$  of a celestial body given the observer's latitude  $L$  and the local hour angle  $LHA$  and declination ( $d$ ) of the body. It thus becomes a replacement for the nine volumes of HO 214. However, the user need not bother with the distinctions of same name and contrary name; the program itself resolves all ambiguities of this type.

**Equations:**

$$H_c = \sin^{-1} [\sin d \sin L + \cos d \cos L \cos LHA]$$

$$Z_n = \begin{cases} Z & ; \sin LHA < 0 \\ 360-Z; \sin LHA \geq 0 & \end{cases} \quad Z = \cos^{-1} \left[ \frac{\sin d - \sin L \sin H_c}{\cos L \cos H_c} \right]$$

### **Remarks:**

Southern latitudes and southern declinations must be entered as negative numbers.

The meridian angle  $t$  may be input in place of LHA, but if so, eastern meridian angles must be input as negative numbers.

This program assumes the calculator is set in DEG mode.

A-000 LBLA		A-030 R 1	
A-001 +H	L.	A-031 SIN	
A-002 S 0		A-032 R 3	
A-003 RTN	-----	A-033 R 0	
A-004 +H		A-034 SIN	
A-005 S 1	d.	A-035 x	
A-006 R/S		A-036 -	
A-007 +H	-----	A-037 R 0	
A-008 S 2	LHA.	A-038 COS	
A-009 R 0		A-039 ÷	
A-010 SIN		A-040 R 4	
A-011 R 1		A-041 COS	
A-012 SIN		A-042 ÷	
A-013 x		A-043 COS	Z
A-014 R 0		A-044 R 2	
A-015 COS		A-045 SIN	
A-016 R 1		A-046 X 0	
A-017 COS		A-047 ST00	
A-018 x		A-048 R4	
A-019 R 2		A-049 3	
A-020 COS		A-050 6	
A-021 x	Hc, dec. deg.	A-051 0	
A-022 +		A-052 X 2 Y	
A-023 S 3		A-053 -	
A-024 SIN		A-054 PRTX	Zn.
A-025 S 4		A-055 RTN	
A-026 +HMS		A-056 LBL0	
A-027 FIX4		A-057 R4	
A-028 PRTX	Hc, D.MS.	A-058 PRTX	Zn.
A-029 FIX1			
<b>REGISTERS</b>			
0 L	1 d	2 LHA	3 sin Hc
4 Hc	5	6	7
8	9	0	.1
.2	.3	.4	.5
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in the following:			
	Observer's latitude	L, D.MS	A	L, dec. deg.

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	Declination	d, D.MS	R/S	d, dec. deg.
	Local hour angle	LHA, D.MS	R/S	Hc, D.MS
				Zn, dec. deg.

**Example:**

Compute the altitude and azimuth of the moon if its LHA is  $2^{\circ}39'54''\text{W}$  and its declination  $13^{\circ}51'06''\text{S}$ . The assumed latitude is  $33^{\circ}20'\text{N}$ .

**Keystrokes:**

33.20 [A] →

13.5106 [CHS] [R/S] →

2.3954 [R/S] →

**Outputs:**

33.33

-13.85

42.4447 \*\*\* (Hc, D.MS)  
183.5 \*\*\* (Zn, dec. deg.)**TIME OF SUNRISE AND SUNSET**

The following program computes the time of sunrise and sunset given the month and day and the observer's latitude and longitude. All times given are Greenwich Mean Time.

**Equations:**

$$\text{Sunrise: } R = [\lambda - \cos^{-1}(-\tan \phi_s \tan L)]/15 - E + 12$$

$$\text{Sunset: } S = [\lambda + \cos^{-1}(-\tan \phi_s \tan L)]/15 - E + 12$$

where

$\lambda$ ,  $\lambda$  = latitude, longitude of observer

$\phi_s$  = subsolar latitude (declination of sun)

$E$  = equation of time

$\phi_s$  and  $E$  are approximated by

$$\phi_s = -23.5 \cos(t + 10)$$

$$E = 0.123 \cos(t + 87) - 1/6 \sin(2t + 20)$$

$$t = 0.988 [D - 1 + 30.3(m - 1)]$$

where D and m are day and month, respectively.

### Remarks:

These equations compute the time at which the middle of the sun is on the horizon. Atmospheric refraction, which would cause the sun to rise earlier and set later, is not accounted for.

Southern latitudes and eastern longitudes are expressed as negative values.

The approximate values of  $\phi_s$  and E cause s to exhibit a maximum error of +4.7 minutes and -0.6 minutes at 45° north latitude, based on 1973 ephemeris data. Refraction and secular changes in the ephemeris can result in errors as large as +8 minutes from observed data at 45° north. Errors decrease as latitudes approach 0°. Large errors exist above 65°.

This program assumes the calculator is set in DEG mode.

A-000 LBLA		A-040 2
A-001 S 0	D.	A-041 0
A-002 RTN		A-042 +
A-003 S 1	-----	A-043 SIN
A-004 R/S	m.	A-044 6
A-005 +H	-----	A-045 ÷
A-006 S 2		A-046 -
A-007 R/S		A-047 1
A-008 +H		A-048 2
A-009 S 3	-----	A-049 -
A-010 R 1	$\lambda$ .	A-050 S 5
A-011 1		A-051 R 4
A-012 -		A-052 1
A-013 3		A-053 0
A-014 0		A-054 +
A-015 .		A-055 COS
A-016 3		A-056 2
A-017 x		A-057 3
A-018 R 0		A-058 *
A-019 +		A-059 5
A-020 1		A-060 x
A-021 -		A-061 TAN
A-022 .		A-062 R 2
A-023 9		A-063 TAN
A-024 6		A-064 x
A-025 6		A-065 COS-1
A-026 x		A-066 S 6
A-027 S 4	t	A-067 R 3
A-028 6		A-068 R/S
A-029 7		A-069 LBL1
A-030 +		A-070 R 3
A-031 COS		A-071 R 6
A-032 .		A-072 -
A-033 1		A-073 1
A-034 2		A-074 5
A-035 3		A-075 ÷
A-036 x		A-076 R 5
A-037 R 4		A-077 -
A-038 ENT†		A-078 X<0
A-039 +		A-079 6S89

Sunrise.

A-080 +HMS		A-093 ÷	
A-081 RTN		A-094 R 5	
A-082 LBL9		A-095 -	
A-083 2		A-096 2	
A-084 4		A-097 4	
A-085 +		A-098 X $\leq$ Y	
A-086 RTN		A-099 GSB8	
A-087 LBL2	----- Sunset.	A-100 X $\leq$ Y	
A-088 R 3		A-101 +HMS	
A-089 R 6		A-102 RTN	
A-090 +		A-103 LBL8	
A-091 1		A-104 -	
A-092 5		A-105 ENT†	
<b>REGISTERS</b>			
0 D	1 m	2 L, dec. deg.	3 $\lambda$ , dec. deg.
4 t	5 E - 12	6 $\Delta\theta$	7
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in date:			
	Day	D	A	D
	Month	m	R/S	m
3	Key in position:			
	Latitude	L, D.MS	R/S	L, dec. deg.
	Longitude	$\lambda$ , D.MS	R/S	$\lambda$ , dec. deg.
4	Compute either or both:			
	Time of sunrise		GSB 1	H.MS
	Time of sunset		GSB 2	H.MS

**Example:**

What time does the sun rise in San Francisco ( $L37^{\circ}49'N$ ,  $\lambda122^{\circ}25'W$ ) on Christmas Day? What time does it set?

**Keystrokes:**

25 A 12 R/S →  
 37.49 R/S 122.25 R/S →  
 GSB 1 →  
 GSB 2 →

**Outputs:**

12.00  
 122.42  
 15.28  
 0.50

The time of sunrise is thus 15:28 GMT or 7:28 AM Pacific Standard Time; sunset is at 00:50 GMT or 4:50 PM PST.

# NUMERICAL METHODS

## SOLUTION TO $f(x) = 0$

This program is designed to find one real root of the equation  $f(x) = 0$ , where  $f(x)$  is a function specified by the user. The user must also specify two initial guesses,  $a$  and  $b$ , which the program will refine by a modified secant method until it finds either an exact root  $x_0$  ( $f(x_0) = 0$ ) or until two successive approximations are nearly indistinguishable. At this point it will stop, displaying the root found.

If the initial guesses  $a$  and  $b$  bracket the root (i.e.,  $f(a) \times f(b) < 0$ ), then convergence is guaranteed if the function is continuous over the interval  $[a,b]$ . There may also be cases in which the program will iterate forever without converging to a root.

The user-defined function  $f(x)$  should be keyed in under LBL B. The value of  $x$  will be in the display (X-register) when the function is entered. The stack and register  $R_0-R_9$  are available for use in defining  $f(x)$ .

### Remarks:

Of the initial guesses  $a$  and  $b$ , your better guess should be used for  $a$ .

As  $a$  and  $b$  are keyed in, their function values will be computed and displayed. This can be of assistance in finding values to bracket a root ( $f(a) \times f(b) < 0$ ).

After a root has been found, key **B** may be pressed to see if the function value is close enough to zero.

A-000 LBLA		A-022 S.5	
A-001 S.1		A-023 R.5	
A-002 B	a → f(a)	A-024 R.3	
A-003 S.2	-----	A-025 -	
A-004 R/S	-----	A-026 R.4	
A-005 S.3	-----	A-027 R.2	
A-006 E	b → f(b)	A-028 ÷	
A-007 S.4	-----	A-029 1	
A-008 R/S	-----	A-030 -	
A-009 Z		A-031 EEX	
A-010 GT01		A-032 CHS	
A-011LBLG	If AC = 0, exit.	A-033 S	
A-012 X=Y		A-034 -	
A-013 GT09		A-035 ÷	c = a +
A-014 LSTX	New B = old A.	A-036 +	(a-b)/(B/A - 1.000000001)
A-015 S.4		A-037 X#Y	If c ≠ a, GTO 2.
A-016 R.5		A-038 GT02	
A-017 S.3	New b = old a.	A-039 R.3	
A-018 Z	-----	A-040 -	
A-019LBLI		A-041 S	
A-020 S.0		A-042 ÷	
A-021 R.1	New p = 2 or 2 × (old p).	A-043 -	c = a - (a-b)/9.

A-044 X#Y A-045 GTO2 A-046 S.1 A-047 R/S A-048 LBL2 A-049 S.1 A-050 B A-051 R.4 A-052 R.2 A-053 x A-054 D A-055 LSTX A-056 RT A-057 S.2 A-058 X#Y A-059 x A-060 X≤Y	If c = a, halt.  ----- If c ≠ a, then save c.  c = f(c)  New A = old C.	A-061 GTO0 A-062 RT A-063 X≤Y A-064 GTO0 A-065 R.0 A-066 R.4 A-067 X#Y A-068 ÷ A-069 S.4 A-070 CL X A-071 LSTX A-072 ENT1 A-073 + A-074 GTO1 A-075 LBL9 A-076 R.1	Else if AB ≥ 0, GTO 0.  If AC > 0 and AB < 0, then new B = old B/p.  New p = 2 x (old p).  Root. -----
<b>REGISTERS</b>			
0	1	2	3
4	5	6	7
8	9	.0	.1 c
.2 A	.3 b	.4 B	.5 a
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Switch to PRGM.		9 B	(CLEAR B)
	Key in function f(x).			
	Switch to RUN.			
3	Key in initial guesses.	a	A	f(a)
		b	R/S	f(b)
4	Compute root.		R/S	Root
5	To find f(x) for any x.	x	B	f(x)

**Example:**

Find a root of the equation  $\ln x + 3x - 10.8074$  in the interval  $[1, 5]$ . Store the constant 10.8074 in  $R_1$ . Find the actual function value of the reported root.

**Keystrokes:**

Switch to PRGM.

9 CLEAR B

f LN f LAST X 3 ×

+ RCL 1 -

**Outputs:**

Switch to RUN.

10.8074	<b>STO</b>	1	A	→	-7.81	(f(1))
5	<b>R/S</b>			→	5.80	(f(5))
	<b>R/S</b>			→	3.21	(Root)
	<b>B</b>			→	0.00	(f(3.21))

## NUMERICAL INTEGRATION, DISCRETE CASE

Let  $x_0, x_1, \dots, x_n$  be equally spaced points  $x_i = x_0 + ih$  for  $i = 1, 2, \dots, n$ , at which corresponding values  $f(x_0), f(x_1), \dots, f(x_n)$  of a function  $f(x)$  are known. Using only this information, with no explicit expression for  $f(x)$  itself,

$$\int_{x_0}^{x_n} f(x) dx$$

may be approximated using

(1) The trapezoidal rule:

$$\int_{x_0}^{x_n} f(x) dx \cong \frac{h}{2} \left[ f(x_0) + 2 \sum_{i=1}^{n-1} f(x_i) + f(x_n) \right] = I_1$$

(2) The Simpson's rule:

$$\begin{aligned} \int_{x_0}^{x_n} f(x) dx &\cong \frac{h}{3} \left[ f(x_0) + 4f(x_1) + 2f(x_2) \right. \\ &\quad \left. + \dots + 4f(x_{n-3}) + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n) \right] = I_2 \end{aligned}$$

In order to apply Simpson's rule,  $n$  must be even.

### Remarks:

Attempting to compute by Simpson's rule with  $n$  odd will cause the error display, SEE 5.

A-000 LBLA		A-022 GT09	-----
A-001 S S	h	A-023 LBL1	Trapezoidal area.
A-002 R/S	-----	A-024 Z	-----
A-003 CL Σ	-----	A-025 R.1	-----
A-004 S.1	-----	A-026 GT00	-----
A-005 S.3	f(x <sub>0</sub> )	A-027 LBL2	Simpson area.
A-006 CL X	-----	A-028 R.0	-----
A-007 LBL9	-----	A-029 Z	-----
A-008 R/S	-----	A-030 ÷	-----
A-009 S 8	f(x <sub>j</sub> ), j odd.	A-031 FRAC	Test n even.
A-010 ENT†	-----	A-032 CHS	-----
A-011 +	-----	A-033 X#0	-----
A-012 ENT†	-----	A-034 TX	-----
A-013 +	-----	A-035 3	-----
A-014 LSTX	2 f(x <sub>j</sub> ), 4 f(x <sub>j</sub> ).	A-036 R.3	-----
A-015 Σ+	-----	A-037 LBL0	-----
A-016 R/S	-----	A-038 R 8	-----
A-017 S 8	-----	A-039 -	-----
A-018 ENT†	f(x <sub>j</sub> ), j even.	A-040 R 9	-----
A-019 +	-----	A-041 X	-----
A-020 ENT†	-----	A-042 XΣY	-----
A-021 Σ+	2 f(x <sub>j</sub> ), 2 f(x <sub>j</sub> ).	A-043 ÷	-----
<b>REGISTERS</b>			
0	1	2	3
4	5	6	7
8 f(x <sub>j</sub> )	9 h	.0 n	.1 Σ TRAP
.2 Used	.3 Σ SIMP	.4 Used	.5 Used
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in spacing.	h	[A]	
3	Key in f(x <sub>0</sub> ).	f(x <sub>0</sub> )	[R/S]	0
4	Repeat step 4 for all function values.	f(x <sub>j</sub> )	[R/S]	j
5	After all function values have been keyed in, calculate either or both:			
	Trapezoidal approx.		[GSB 1]	TRAP
	Simpson's rule.		[GSB 2]	SIMP

**Example:**

Given the values below for  $f(x_j)$ ,  $j = 0, 1, \dots, 8$ , compute the approximations to the integral

$$\int_0^2 f(x) dx$$

by the trapezoidal rule and by Simpson's rule.

The value for  $h$  is 0.25.

i	0	1	2	3	4	5	6	7	8
$x_i$	0	.25	.5	.75	1	1.25	1.5	1.75	2
$f(x_i)$	2	2.8	3.8	5.2	7	9.2	12.1	15.6	20

**Keystrokes:**

.25 [A] 2 [R/S] 2.8 [R/S]  
 3.8 [R/S] 5.2 [R/S] 7 [R/S]  
 9.2 [R/S] 12.1 [R/S] 15.6 [R/S]  
 20 [R/S] →                          8.00  
 [GSB] [1] →                          16.68  
 [GSB] [2] →                          16.58

**Outputs:**

(n)  
 (Trapezoidal)  
 (Simpson's)

## NUMERICAL INTEGRATION, SIMPSON'S RULE

This program approximates the definite integral  $\int_a^b f(x) dx$  by Simpson's rule when a formula for  $f(x)$  is known explicitly. The user must specify the endpoints  $a$  and  $b$  of the interval over which integration is to be performed and the number of subintervals  $n$  into which the interval  $(a, b)$  is to be divided. This  $n$  must be even; if it is not, the error message "See 5" will be displayed. The program will go on to compute  $x_0 = a$ ,  $x_j = x_0 + jh$  for  $j = 1, 2, \dots, n-1$ , and  $x_n = b$  where

$$h = \frac{b - a}{n}$$

The integral  $\int_a^b f(x) dx$  is approximated by Simpson's rule:

$$\int_{x_0}^{x_n} f(x) dx \cong \frac{h}{3} \left[ f(x_0) + 4f(x_1) + 2f(x_2) \right. \\ \left. + \dots + 4f(x_{n-3}) + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n) \right]$$

The program assumes that  $f(x)$  will be keyed into program memory under LBL B. Registers  $R_0-R_9$  and the stack are available for use in defining  $f(x)$ . The routine  $f(x)$  will find the value of  $x$  in the X-register upon entry. Two levels of subroutines are allowed in defining  $f(x)$ .

A-000 LBL A A-001 S.3 A-002 X#Y A-003 S.2 A-004 R/S A-005 S I A-006 ÷ A-007 ÷ A-008 FRAC A-009 CHS A-010 X#0 A-011 JX A-012 R.2 A-013 B A-014 S.1 A-015 R.3 A-016 B A-017 R.1 A-018 + A-019 S.1	b  a  n  Display "See 5" if n is not even.  R.1 $\leftarrow f(a) + f(b)$ .	A-020 R.3 A-021 R.2 A-022 S.5 A-023 - A-024 I A-025 ÷ A-026 S.4 A-027 ÷ A-028 S.0 A-029 DSZ A-030 LBL B A-031 R.4 A-032 R.5 A-033 + A-034 S.5 A-035 B A-036 6 A-037 R.0 A-038 - A-039 S.0	$x_0 \leftarrow a$  $h \leftarrow (b - a)/n$ .  $R_0$ will be 2 or 4.  -----  $x \leftarrow x + h$ $f(x)$
--	--	--	--

A-040 x A-041 R.1 A-042 + A-043 S.1 A-044 DSZ A-045 GT09	R. <sub>1</sub> ← R. <sub>1</sub> + 4 (or 2) f(x).	A-046 R.1 A-047 R.4 A-048 x A-049 3 A-050 ÷	Area.
<b>REGISTERS</b>			
0	1	2	3
4	5	6	7
8	9	.0 4, 2	.1 Σ Simp
.2 a	.3 b	.4 h	.5 x I n

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Switch to PRGM.		<b>a</b> <b>B</b>	(CLEAR B)
	Key in f(x).			
	Switch to RUN.			
3	Key in endpoints.	a	<b>ENTER</b>	
		b	<b>A</b>	a
4	Key in number of sub-intervals.	n	<b>R/S</b>	Area

**Example:**

Find the value of  $\int_0^{2\pi} \frac{dx}{1 - \cos x + 0.25}$  for u = 16. Note that x is assumed

to be in radians. For safety, if you work mostly in degrees, it is good programming practice to set the angular mode to radians at the beginning of the routine, then back to degrees at the end.

**Keystrokes:**

Switch to PRGM.

**g** CLEAR **B**

**g** **RAD** **f** **COS** 1

**x<sub>2</sub>y** **-** .25 **+**

**1/x** **g** **DEG**

**Outputs:**

Switch to RUN.

0 [ENTER]	2 9 [π] × [A]	→	0.00
16 [R/S]	→	8.36	(Area)

The exact solution is  $\frac{8\pi}{3} = 8.38$ .

## DIFFERENTIAL EQUATIONS

This program solves first-order differential equations by the fourth-order Runge-Kutta method. A first-order equation is of the form  $y' = f(x,y)$  with initial values  $(x_0, y_0)$ . The solution is a numerical solution which calculates  $y_i$  for  $x_i = x_0 + ih$ ,  $i = 1, 2, 3, \dots$ , where  $h$  is an increment specified by the user.

The user must specify the function  $f(x,y)$  by keying it into program memory under LBL B. The stack and registers  $R_0-R_9$  and I are available for defining  $f(x,y)$ . The function should assume that  $x$  and  $y$  will be in the X- and Y-registers, respectively, upon entry.

### Equations:

$$y_{i+1} = y_i + \frac{1}{6} (c_1 + 2c_2 + 2c_3 + c_4)$$

where

$$c_1 = hf(x, y_i)$$

$$c_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{c_1}{2}\right)$$

$$c_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{c_2}{2}\right)$$

$$c_4 = hf(x_i + h, y_i + c_3)$$

A-000 LBLA		A-029 LSTX	
A-001 ÷		A-030 ENT↑	
A-002 ÷		A-031 +	
A-003 S.4	h/2	A-032 R.3	$y_i + c_3$
A-004 R/S		A-033 +	
A-005 S.3	$y_0$	A-034 R.2	
A-006 X $\leftrightarrow$ Y		A-035 R.4	
A-007 S.2	$x_0$	A-036 +	
A-008 SPC	-----	A-037 S.2	
A-009LBL9		A-038 B	$x_i + h \rightarrow x_{i+1}$
A-010 B		A-039 R.1	
A-011 S.1	$c_1/2$	A-040 +	
A-012 R.3		A-041 R.0	
A-013 +	$y_i + (c_1/2)$	A-042 ENT↑	
A-014 R.2		A-043 +	
A-015 R.4		A-044 +	
A-016 +		A-045 3	$\Delta = \frac{1}{6} (c_1 + 2c_2 + 2c_3 + c_4)$
A-017 S.2	$x_i + (h/2)$	A-046 ÷	
A-018 B	$c_2/2$	A-047 R.3	
A-019 S.0		A-048 +	
A-020 R.3		A-049 S.3	
A-021 +	$y_i + (c_2/2)$	A-050 R.2	
A-022 R.2	$x_i + (h/2)$	A-051 PRTX	
A-023 B		A-052 R.3	$x_{i+1}$
A-024 R.0		A-053 PRTX	
A-025 X $\leftrightarrow$ Y		A-054 SPC	
A-026 +		A-055 X $\leftrightarrow$ Y	$y_{i+1}$
A-027 S.0	$c_3/2$	A-056 GT09	
A-028 CL X			
<b>REGISTERS</b>			
0	1	2	3
4	5	6	7
8	9	.0 $c_2 + c_3$	.1 $c_1$
.2 $x_i$	.3 $y_i$	.4 h/2	.5

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	<b>To key in f(x):</b>			
	Switch to PRGM.		<b>9 B</b>	(CLEAR B)
	Key in function and f(x), end with		<b>RCL</b> <b>□</b>	
			<b>4 x</b>	
	Switch to RUN.			
3	Key in spacing.	$h$	<b>A</b>	$h/2$
4	Key in initial x and y values			
	and output succeeding values.	$x_0$	<b>ENTER</b>	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
		$y_0$	R/S	$x_i$
				$y_i$

**Example:**

Solve numerically the first-order differential equation

$$y' = \frac{\sin x + \tan^{-1}(y/x)}{y - \ln(\sqrt{x^2 + y^2})}$$

where  $x_0 = y_0 = 1$ . Let  $h = 0.5$ . The angular mode must be set to radians.

**Keystrokes:**

Switch to PRGM.

```
g CLEAR B
g RAD STO 1 x:y
STO 2 x:y g P
f LN STO 3 R+
RCL 1 f SIN +
RCL 2 RCL 3 - +
g DEG RCL 4 x
```

**Outputs:**

Switch to RUN.

.5 A → 0.25 (h/2)

1 ENTER + 1 R/S →	1.50 *** ( $x_1$ )
	2.06 *** ( $y_1$ )
	2.00 *** ( $x_2$ )
	2.78 *** ( $y_2$ )
	2.50 *** ( $x_3$ )
	3.28 *** ( $y_3$ )

R/S (halts program)

# STATISTICS

## CURVE FITTING

This program performs curve fits by three different methods: exponential, logarithmic, and power. The linear curve fit (linear regression) is already a preprogrammed function so is not included here. For each curve fit, the data points are entered by keying in the y-value, pressing **ENTER**, and keying in the x-value. After all (x,y) data pairs have been keyed in, the following calculations may be made: regression constants, coefficient of determination, estimate of y ( $\hat{y}$ ), and estimate of x ( $\hat{x}$ ).

### **Equations:**

#### *Exponential*

The exponential equation  $y = a e^{bx}$  is fit by applying linear regression to the equation

$$\ln y = bx + \ln a.$$

All y-values must be positive.

$$b = \frac{\sum x_i \ln y_i - \frac{1}{n} (\sum x_i)(\sum \ln y_i)}{\sum x_i^2 - \frac{1}{n} (\sum x_i)^2}$$

$$a = \exp \left[ \frac{\sum \ln y_i}{n} - b \frac{\sum x_i}{n} \right]$$

$$\hat{y} = ae^{bx}$$

$$\hat{x} = \frac{1}{b}(\ln y - \ln a)$$

#### *Logarithmic*

The logarithmic equation is  $y = a + b \ln x$ . In this curve fit, all x-values must be positive.

$$b = \frac{\sum y_i \ln x_i - \frac{1}{n} \sum \ln x_i \sum y_i}{\sum (\ln x_i)^2 - \frac{1}{n} (\sum \ln x_i)^2}$$

$$a = \frac{1}{n} (\sum y_i - b \sum \ln x_i)$$

$$\hat{y} = a + b (\ln x)$$

$$\hat{x} = e^{[(y - a)/b]}$$

### *Power*

A power curve is of the form  $y = ax^b$ . This is linearized to the form

$$\ln y = \ln a + b (\ln x).$$

Here both x- and y-values must be positive.

$$b = \frac{\sum (\ln x_i)(\ln y_i) - \frac{(\sum \ln x_i)(\sum \ln y_i)}{n}}{\sum (\ln x_i)^2 - \frac{(\sum \ln x_i)^2}{n}}$$

$$a = \exp \left[ \frac{\sum \ln y_i}{n} - b \frac{\sum \ln x_i}{n} \right]$$

$$\hat{y} = ax^b$$

$$\hat{x} = e^{[(\ln y - \ln a)/b]}$$

### *Coefficient of determination*

For all cases, the value of  $r^2$  is computed as shown below:

$$r^2 = \left( b \frac{s_x}{s_y} \right)^2$$

where  $s_x, s_y$  = standard deviations of x and y.

### **Remarks:**

Each curve fit routine (under LBL's B, C, and D) is completely self-contained and, if desired, may be keyed into program memory without any of the other routines.

A-000 LBL4		C-014 LSTX	
A-001 CL $\Sigma$	Clear.	C-015 $x$	
B-000 LBL5		C-016 $x \neq y$	
B-001 $x \neq y$	Exponential.	C-017 $\div$	
B-002 LN		C-018 $x^2$	
B-003 $x \neq y$		C-019 RTN	
B-004 $\Sigma +$		C-020 LBL3	
B-005 RTN		C-021 LN	
B-006 LBL1		C-022 $\hat{v}$	$r^2$
B-007 LR		C-023 RTN	
B-008 $e^x$		C-024 LBL4	
B-009 PRTX	a	C-025 LR	
B-010 $x \neq y$	b	C-026 LSTX	
B-011 PRTX		C-027 $x \neq y$	
B-012 RTN		C-028 $-$	
B-013 LBL2		C-029 $x \neq y$	
B-014 LR		C-030 $\div$	
B-015 $x \neq y$		C-031 $e^x$	
B-016 S		D-000 LBLD	
B-017 LSTX		D-001 LN	
B-018 $x$		D-002 $x \neq y$	
B-019 $x \neq y$		D-003 LN	
B-020 $\div$		D-004 $x \neq y$	
B-021 $x^2$	$r^2$	D-005 $\Sigma +$	
B-022 RTN		D-006 RTN	
B-023 LBL3		D-007 LBL1	
B-024 $\hat{v}$		D-008 LR	
B-025 $e^x$		D-009 $e^x$	a
B-026 RTN		D-010 PRTX	b
B-027 LBL4		D-011 $x \neq y$	
B-028 LR		D-012 PRTX	
B-029 LSTX		D-013 RTN	
B-030 LN		D-014 LBL2	
B-031 $x \neq y$		D-015 LR	
B-032 $-$		D-016 $x \neq y$	
B-033 $x \neq y$		D-017 S	
B-034 $\div$		D-018 LSTX	
B-035 RTN		D-019 $x$	
C-000 LBLC	Logarithmic.	D-020 $x \neq y$	
C-001 LN		D-021 $\div$	$r^2$
C-002 $\Sigma +$		D-022 $x^2$	
C-003 RTN		D-023 RTN	
C-004 LBL1		D-024 LBL3	
C-005 LR	a	D-025 LN	
C-006 PRTX	b	D-026 $\hat{v}$	
C-007 $x \neq y$		D-027 $e^x$	
C-008 PRTX		D-028 RTN	
C-009 RTN		D-029 LBL4	
C-010 LBL2		D-030 LN	
C-011 LR		D-031 R $\downarrow$	
C-012 $x \neq y$		D-032 LR	
C-013 S			

<i>D-033 R↑</i>			<i>D-036 X↑Y</i>		
<i>D-034 X↑Y</i>			<i>D-037 ÷</i>		
<i>D-035 -</i>			<i>D-038 e^x</i>		
<b>REGISTERS</b>					
0	1	2	3		
4	5	6	7		
8	9	.0 n	.1 $\Sigma x$		
.2 $\Sigma x^2$	.3 $\Sigma y$	.4 $\Sigma y^2$	.5 $\Sigma xy$	I	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	
3	Key in y-value and x-value.	$y_i$	ENTER ↴	
		$x_i$		
4	Select one of curve fits below:			
	Exponential		B	i
	Logarithmic		C	i
	Power		D	i
5	Repeat steps 3 and 4 for all data pairs.			
6	Find regression constants.		GSB [1]	a
				b
7	Find coefficient of determination.		GSB [2]	$r^2$
8	For a given x-value, find an estimated y.	x	GSB [3]	$\hat{y}$
9	For a given y-value, find an estimated x.	y	GSB [4]	$\hat{x}$
10	For a new curve fit, go to step 2.			

**Example 1:**

Determine whether a power curve or an exponential curve is the better fit to the data below:

x	16	27	33	41	56
y	23	30	35	38	42

**Keystrokes:****A**

23 [ENTER+] 16 [D] → 1.00

30 [ENTER+] 27 [D] → 2.00

35 [ENTER+] 33 [D] → 3.00

38 [ENTER+] 41 [D] → 4.00

42 [ENTER+] 56 [D] → 5.00

GSB [1] → 5.93 \*\*\* (a)  
0.50 \*\*\* (b)Thus the power curve is  $y = 5.93 x^{0.50}$ .GSB [2] → 0.98 ( $r^2$ )**A** (Start exponential)

23 [ENTER+] 16 [B] → 1.00

30 [ENTER+] 27 [B] → 2.00

35 [ENTER+] 33 [B] → 3.00

38 [ENTER+] 41 [B] → 4.00

42 [ENTER+] 56 [B] → 5.00

GSB [1] → 19.71 \*\*\* (a)  
0.01 \*\*\* (b)Thus the exponential curve is  $y = 19.71 e^{0.01x}$ .GSB [2] → 0.90 ( $r^2$ )

Since  $r^2$  for the power curve  $0.98 > 0.90$ , we conclude that the power curve fits the data better than the exponential curve.

**Example 2:**

A manufacturer observes declining sales of a soon-to-be-obsolete product, of which there were originally 10,000 units in inventory. The cumulative sales

figures for a number of months are shown below. If these are fit by a logarithmic curve, find the projected cumulative sales after 7 months. Find also the month in which the last unit (number 10,000) is projected to be sold.

Month (x)	1	2	3	4	5	6
Cumulative Sales (y)	1431	3506	5177	6658	7810	8592

**Keystrokes:****A**1431 **ENTER** 1 **C** → 1.003506 **ENTER** 2 **C** → 2.005177 **ENTER** 3 **C** → 3.006658 **ENTER** 4 **C** → 4.007810 **ENTER** 5 **C** → 5.008592 **ENTER** 6 **C** → 6.00**GSB** 1 → 1066.15 \*\*\* (a)  
4069.93 \*\*\* (b)**GSB** 2 → 0.99 ( $r^2$ )7 **GSB** 3 → 8985.87 ( $x \rightarrow \hat{y}$ )

At the end of 7 months, the cumulative sales are expected to total 8986 units.

10000 **GSB** 4 → 8.98 ( $y \rightarrow \hat{x}$ )**Outputs:**

(Start logarithmic)

The final unit is projected to be sold near the end of the ninth month.

## COVARIANCE AND CORRELATION COEFFICIENT

For a set of given data points  $\{(x_i, y_i), i = 1, 2, \dots, n\}$ , the covariance and correlation coefficient are found by

$$\text{covariance } s_{xy} = \frac{1}{n - 1} \left( \sum x_i y_i - \frac{1}{n} \sum x_i \sum y_i \right)$$

$$\text{or } s_{xy}' = \frac{1}{n} \left( \sum x_i y_i - \frac{1}{n} \sum x_i \sum y_i \right)$$

$$\text{correlation coefficient } r = \frac{B s_x}{s_y}$$

where  $B$  = slope from linear regression

$s_x, s_y$  = standard deviations of  $x$  and  $y$

### Remarks:

The correlation coefficient  $r$  always lies in the range  $-1 \leq r \leq 1$ . It may also be found by

$$r = \frac{s_{xy}}{s_x s_y}$$

A-000 LBL A		A-015 ÷	s <sub>xy</sub> .
A-001 CL Σ	Clear.	A-016 RTN	-----
A-002 RTN	-----	A-017 R 0	
A-003LBL1	-----	A-018 ×	
A-004 R.5	-----	A-019 R.0	
A-005 R.1	-----	A-020 ÷	
A-006 R.3	-----	A-021 RTN	
A-007 ×	-----	A-022LBL2	
A-008 R.0	-----	A-023 LR	
A-009 ÷	-----	A-024 X <sup>2</sup> Y	
A-010 -	-----	A-025 S	
A-011 R.0	-----	A-026 LSTX	
A-012 1	-----	A-027 X	
A-013 -	-----	A-028 X <sup>2</sup> Y	
A-014 S 0	-----	A-029 ÷	
<b>REGISTERS</b>			
0 n - 1	1	2	3
4	5	6	7
8	9	.0 n	.1 Σx
.2 Σx <sup>2</sup>	.3 Σy	.4 Σy <sup>2</sup>	.5 Σxy
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	
3	Key in y-value and x-value.	$y_i$	ENTER	
		$x_i$	$\Sigma+$	i
4	Repeat step 3 for all data pairs.			
5	Find covariance.		GSB [1]	$s_{xy}$
			R/S	$s_{xy}'$
6	Find correlation coefficient.		GSB [2]	r
7	For a new case go to step 2.			

**Example:**Find  $s_{xy}$ ,  $s_{xy}'$ , and r for the data below:

x	26	30	44	50	62
y	92	85	78	81	54

**Keystrokes:**

A

- 92 ENTER  $\Sigma+$  → 1.00
- 85 ENTER  $\Sigma+$  → 2.00
- 78 ENTER  $\Sigma+$  → 3.00
- 81 ENTER  $\Sigma+$  → 4.00
- 54 ENTER  $\Sigma+$  → 5.00
- GSB [1] → -191.00 ( $s_{xy}$ )
- R/S → -152.80 ( $s_{xy}'$ )
- GSB [2] → -0.90 (r)

**Outputs:**

## MOMENTS AND SKEWNESS

This program computes the first three moments and the moment coefficient of skewness for a set of data  $\{x_1, x_2, \dots, x_n\}$ .

### Equations:

$$\text{1}^{\text{st}} \text{ moment} \quad \bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\text{2}^{\text{nd}} \text{ moment} \quad m_2 = \frac{1}{n} \sum x_i^2 - \bar{x}^2$$

$$\text{3}^{\text{rd}} \text{ moment} \quad m_3 = \frac{1}{n} \sum x_i^3 - \frac{3}{n} \bar{x} \sum x_i^2 + 2\bar{x}^3$$

moment coefficient of skewness

$$\gamma_1 = \frac{m_3}{m_2^{3/2}}$$

H-000 LBLH A-001 CL Σ B-000LBLB B-001 ENT↑ B-002 X² B-003 Σ+ B-004 RTN B-005LBL1 B-006 X̄ B-007 XΣY B-008 S 0 B-009 RTN B-010 R.1 B-011 R.0 B-012 ÷ B-013 R 0 B-014 X² B-015 - B-016 S 1 B-017 RTN B-018 R.5 B-019 R.0	Clear. ----- Sum x, x². ----- $\bar{x}$ -----	B-020 ÷ B-021 R.1 B-022 R 0 B-023 X B-024 R.0 B-025 ÷ $\bar{x}$ B-027 X B-028 - B-029 R 0 B-030 ENT↑ B-031 X² B-032 X B-033 Σ B-034 X B-035 + B-036 S 2 B-037 RTN B-038 R 2 B-039 R 1 B-040 1 B-041 .	$m_3$ -----
---	--	--	----------------

B-042 B-043	5 $\gamma_1$			E-044	$\div$	$\gamma_1$	---
REGISTERS							
0 $\bar{x}$	1 $m_2$	2 $m_3$	3				
4	5	6	7				
8	9	.0 $n$	.1 $\Sigma x^2$				
.2 $\Sigma x^4$	.3 $\Sigma x$	.4 $\Sigma x^2$	.5 $\Sigma x^3$	I			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	
3	Key in $x$ -value.	$x_i$	B	i
4	Repeat step 3 for all $x$ -values.			
5	Compute the moments and skewness.		GSB [1]	$\bar{x}$
			R/S	$m_2$
			R/S	$m_3$
			R/S	$\gamma_1$

**Example:**

Compute the statistics of moment and skewness for the  $x$ -values below:

i	1	2	3	4	5	6
$x_i$	2.1	3.5	4.2	6.5	4.1	3.6

**Keystrokes:**

- A 2.1 B →
- 3.5 B 4.2 B →
- 6.5 B 4.1 B 3.6 B →
- GSB [1] →
- R/S →
- R/S →
- R/S →

**Outputs:**

- |      |              |
|------|--------------|
| 1.00 | (x̄)         |
| 3.00 | $(m_2)$      |
| 6.00 | $(m_3)$      |
| 4.00 | $(\gamma_1)$ |
| 1.72 | $(m_2)$      |
| 1.43 | $(m_3)$      |
| 0.63 | $(\gamma_1)$ |

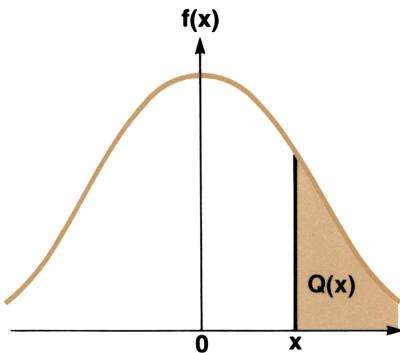
## NORMAL DISTRIBUTION

The density function for a standard normal variable is

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}.$$

The upper tail area is

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{t^2}{2}} dt.$$



For  $x \geq 0$ , polynomial approximation is used to compute  $Q(x)$ :

$$Q(x) = f(x)(b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5) + \epsilon(x)$$

where  $|\epsilon(x)| < 7.5 \times 10^{-8}$

$$t = \frac{1}{1 + rx}, \quad r = 0.2316419$$

$$b_1 = .31938153, \quad b_2 = -.356563782$$

$$b_3 = 1.781477937, \quad b_4 = -1.821255978$$

$$b_5 = 1.330274429$$

### Remarks:

The program only works for  $x \geq 0$ . Equations  $f(-x) = f(x)$ ,  $Q(-x) = 1 - Q(x)$ , where  $x \geq 0$ , can be used to find  $f$  and  $Q$  for negative numbers.

**Reference:**

Abramowitz and Stegun, *Handbook of Mathematical Functions*, National Bureau of Standards, 1968.

A-000 LBLA	x	A-020 1/X	t
A-001 S 6		A-021 ENT↑	
A-002 ENT↑		A-022 ENT↓	
A-003 ×		A-023 ENT1	
A-004 ÷		A-024 R 5	
A-005 ±		A-025 x	
A-006 CHS		A-026 R 4	
A-007 e <sup>x</sup>		A-027 +	
A-008 1 <sup>x</sup>		A-028 -	
A-009 2 <sup>x</sup>		A-029 R 3	
A-010 x <sup>2</sup>	f(x)	A-030 +	
A-011 √x	-----	A-031 x	
A-012 ÷		A-032 R 2	
A-013 S 7		A-033 +	
A-014 R/S		A-034 x	
A-015 R 0		A-035 R 1	
A-016 R 6		A-036 +	
A-017 x		A-037 x	
A-018 1		A-038 R 7	
A-019 +		A-039 x	Q(x)
<b>REGISTERS</b>			
0 r	1 b <sub>1</sub>	2 b <sub>2</sub>	3 b <sub>3</sub>
4 b <sub>4</sub>	5 b <sub>5</sub>	6 x	7 f(x)
8	9	.0	.1
.2	.3	.4	.5
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store constants.	r	STO [0]	
		b <sub>1</sub>	STO [1]	
		b <sub>2</sub>	STO [2]	
		b <sub>3</sub>	STO [3]	
		b <sub>4</sub>	STO [4]	
		b <sub>5</sub>	STO [5]	
3	Key in x and compute f(x) and			
	Q(x).	x	[A]	f(x)
			[R/S]	Q(x)
4	For a new x, go to step 3.			

**Example:**

Find  $f(x)$  and  $Q(x)$  for  $x = 1.18$  and  $x = 2.28$ .

**Keystrokes:**

.2316419 **STO** **0** →  
 .31938153 **STO** **0** →  
 .356563782 **CHS** **STO** **2** →  
 1.781477937 **STO** **3** →  
 1.821255978 **CHS** **STO** **4** →  
 1.330274429 **STO** **5** →  
 1.18 **A** →  
**R/S** →  
 2.28 **A** →  
**R/S** →

**Outputs:**

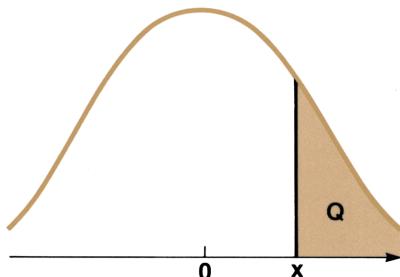
0.23	(r)
0.32	(b <sub>1</sub> )
-0.36	(b <sub>2</sub> )
1.78	(b <sub>3</sub> )
-1.82	(b <sub>4</sub> )
1.33	(b <sub>5</sub> )
0.20	(f(x))
0.12	(Q(x))
0.03	(f(x))
0.01	(Q(x))

**INVERSE NORMAL INTEGRAL**

This program determines the value of  $x$  such that

$$Q = \int_x^{\infty} \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}} dt$$

where  $Q$  is given and  $0 < Q \leq 0.5$ .



The following rational approximation is used:

$$x = t - \frac{c_0 + c_1 t + c_2 t^2}{1 + d_1 t + d_2 t^2 + d_3 t^3} + \epsilon(Q)$$

where  $|\epsilon(Q)| < 4.5 \times 10^{-4}$

$$t = \sqrt{\ln \frac{1}{Q^2}}$$

$$c_0 = 2.515517 \quad d_1 = 1.432788$$

$$c_1 = 0.802853 \quad d_2 = 0.189269$$

$$c_2 = 0.010328 \quad d_3 = 0.001308$$

### Reference:

Abramowitz and Stegun, *Handbook of Mathematical Functions*, National Bureau of Standards, 1968.

A-000 LBL6	Q(x) t	A-016 +		
A-001 ENT†		A-017 x		
A-002 x		A-018 1		
A-003 1/X		A-019 +		
A-004 LN		A-020 S 7		
A-005 IX		A-021 CL X		
A-006 S 6		A-022 R 2		
A-007 ENT†		A-023 x		
A-008 ENT†		A-024 R 1		
A-009 ENT†		A-025 +		
A-010 R 5		A-026 x		
A-011 x		A-027 R 0		
A-012 R 4		A-028 +		
A-013 +		A-029 R 7		
A-014 x		A-030 ÷	x	
A-015 R 3		A-031 -		
<hr/>				
REGISTERS				
0 c <sub>0</sub>	1 c <sub>1</sub>	2 c <sub>2</sub>	3 d <sub>1</sub>	
4 d <sub>2</sub>	5 d <sub>3</sub>	6 t	7 Used	
8	9	.0	.1	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store constants.	c <sub>0</sub>	STO [0]	
		c <sub>1</sub>	STO [1]	
		c <sub>2</sub>	STO [2]	
		d <sub>1</sub>	STO [3]	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
		$d_2$	STO 4	
		$d_3$	STO 5	
3	Key in Q and find x.	Q	A	x

**Example:**

Find the x-values corresponding to  $Q = 0.12$  and  $Q = 0.05$ .

**Keystrokes:**

2.515517	STO 0	→	2.52	(c <sub>0</sub> )
0.802853	STO 1	→	0.80	(c <sub>1</sub> )
0.010328	STO 2	→	0.01	(c <sub>2</sub> )
1.432788	STO 3	→	1.43	(d <sub>1</sub> )
0.189269	STO 4	→	0.19	(d <sub>2</sub> )
0.001308	STO 5	→	1.308000000-03	(d <sub>3</sub> )
.12	A	→	1.18	(x)
.05	A	→	1.65	(x)

**Outputs:****PERMUTATION AND COMBINATION**

A permutation is an ordered subset of a set of distinct objects. The number of possible permutations, each containing n objects, that can be formed from a collection of m distinct objects is given by

$${}_m P_n = \frac{m!}{(m - n)!} = m(m - 1) \dots (m - n + 1)$$

where m, n are integers and  $0 \leq n \leq m$ .

A combination is a selection of one or more of a set of distinct objects without regard to order. The number of possible combinations, each containing n objects, that can be formed from a collection of m distinct objects is given by

$${}_m C_n = \frac{m!}{(m - n)! n!} = \frac{m(m - 1) \dots (m - n + 1)}{1 \cdot 2 \cdot \dots \cdot n}$$

where m, n are integers and  $0 \leq n \leq m$ .

This program computes  ${}_m C_n$  using the following algorithm:

1. If  $n \leq m - n$

$${}_m C_n = \frac{m - n + 1}{1} \cdot \frac{m - n + 2}{2} \cdot \dots \cdot \frac{m}{n} .$$

2. If  $n > m - n$ , program computes  ${}_m C_{m-n}$ .

### Remarks:

${}_m P_n$  can also be denoted by  $P_n^m$ ,  $P(m, n)$  or  $(m)_n$ .

${}_m P_0 = 1$ ,  ${}_m P_1 = m$ ,  ${}_m P_m = m!$

${}_m C_n$ , which is also called the binomial coefficient, can be denoted by  $C_n^m$ ,  $C(m, n)$ , or  $\binom{m}{n}$ .

$${}_m C_n = {}_m C_{m-n}$$

$${}_m C_0 = {}_m C_m = 1$$

$${}_m C_1 = {}_m C_{m-1} = m$$

A-000 LBLA	$m \uparrow n \rightarrow {}_m P_n$	A-031 R↓	${}_m P_m = m!$
A-001 X=Y		A-032 RTN	$m \uparrow n \rightarrow {}_m C_n$
A-002 GT05		A-033LBL5	
A-003 X>Y		A-034 N!	
A-004 GT02		B-000LBLB	
A-005 X=0		B-001 X>Y	
A-006 GT03		B-002 GT02	
A-007 1		B-003 -	
A-008 X=Y		B-004 LSTX	
A-009 GT04		B-005 X>Y	
A-010 -		B-006 X>Y	
A-011 S I		B-007 S 9	Store max (n, m-n)
A-012 R↓		B-008 1	Let $m - n > n$
A-013 S 9	$m(m-1) \dots (m - n + 1)$	B-009 S 8	$m - n + 1 \rightarrow R_7$
A-014LBL1		B-010 +	
A-015 R 9		B-011 S 7	
A-016 1		B-012 R↓	
A-017 -		B-013 X=0	
A-018 S 9		B-014 GT03	
A-019 X		B-015LBL5	
A-020 DSZ		B-016 1	
A-021 GT01		B-017 R 8	
A-022 RTN		B-018 +	
A-023LBL2	$n > m:$ error	B-019 S 8	
A-024 0		B-020 X>Y	
A-025 ÷		B-021 GT06	If $j > n$ , exit
A-026LBL3	$mP_0 = 1$	B-022 R 9	
A-027 1		B-023 +	
A-028 RTN		B-024 R 8	
A-029LBL4	$mP_1 = m$	B-025 ÷	$\frac{m - n + j}{j}$
A-030 R4		B-026 Sx7	

B-027 R↓ B-028 GT05 B-029 LBL6 B-030 R 7 B-031 RTN	$m C_n$	B-032 LBL2 B-033 0 B-034 ÷ B-035 LBL3 B-036 1	n > m: error $m C_0$
<b>REGISTERS</b>			
0	1	2	3
4	5	6	7 II
8 j	9 $m - n$	.0	.1
.2	.3	.4	.5 I Used

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in m and n.	m	<b>ENTER ↴</b>	
		n		
3	Find either:			
	Permutations		<b>A</b>	$m P_n$
	Combinations		<b>B</b>	$m C_n$

**Examples:**Find  ${}_{43}P_3$  and  ${}_{43}C_3$ .**Keystrokes:**

43	<b>ENTER ↴</b>	3	<b>A</b>	→	74046.00	$({}_{43}P_3)$
43	<b>ENTER ↴</b>	3	<b>B</b>	→	12341.00	$({}_{43}C_3)$

**Outputs:****RANDOM NUMBER GENERATOR**

This program calculates uniformly distributed pseudo random numbers  $u_i$  in the range

$$0 \leq u_i \leq 1$$

using the multiplicative linear congruential method:

$$u_{i+1} = \text{Fractional part of } (997 u_i)$$

where  $i = 0, 1, 2, \dots$

The period of this generator is 500,000 numbers and the generator passes the frequency test (chi square) for uniformity, the serial test and the run test. The most significant digits (the left hand digits) are the most random; the rightmost digits are significantly less random.

The seed used in the example is a good one. Other seeds may be selected but the quotient of ( $\text{seed} \times 10^7$ ) divided by two or five must not be an integer. Also, it would be wise to statistically test other seeds before using them.

<i>A-000 LBLH</i>	<i>u<sub>0</sub></i>	<i>A-005 9</i> <i>A-006 7</i> <i>A-007 x</i> <i>A-008 FRAC</i> <i>A-009 GTOA</i>	$u_{i+1} = \text{Frac}(997 u_i)$
<b>REGISTERS</b>			
0 Used	1	2	3
4	5	6	7

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Input seed.	$u_0$	[A]	$u_0$
3	Generate random number.		[R/S]	$u_i$
4	Repeat step 3 as often as desired.			

### Example:

Generate random numbers from the seed 0.5284163.

#### Keystrokes:

.5284163 [A] →  
 R/S →  
 R/S →  
 R/S →  
 R/S →

#### Outputs:

0.53	$(u_0)$
0.83	$(u_1)$
0.56	$(u_2)$
0.27	$(u_3)$
0.04	$(u_4)$

## CHI-SQUARE EVALUATION

This program calculates the value of the  $\chi^2$  statistic for the goodness of fit test by the equation

$$\chi^2 = \sum_{i=1}^n \frac{(O_i - E_i)^2}{E_i}$$

where       $O_i$  = observed frequency

$E_i$  = expected frequency

If the expected values are equal

$$\left( E = E_i = \frac{\Sigma O_i}{n} \text{ for all } i \right)$$

then

$$\chi^2 = \frac{n \Sigma O_i^2}{\Sigma O_i} - \Sigma O_i$$

### Remarks:

In order to apply the goodness of fit test to a set of given data, combining some classes may be necessary to make sure that each expected frequency is not too small (say, not less than 5).

A-000 LBLA		A-018 RTN	$\chi^2$
A-001 C		A-019 LBL2	-----
A-002 S 7		A-020 S+8	
A-003 S 8		A-021 X <sup>2</sup>	
A-004 S 9		A-022 S+9	
A-005 RTN		A-023 1	
A-006LBL1		A-024 S+7	
A-007 S 8		A-025 R 7	
A-008 -		A-026 RTN	
A-009 X <sup>2</sup>		A-027 R 7	
A-010 R 8		A-028 R 9	
A-011 -		A-029 X	
A-012 S+9		A-030 R 8	
A-013 1		A-031 -	
A-014 S+7		A-032 R 8	
A-015 R 7	i	A-033 -	$\chi^2$
A-016 RTN		A-034 RTN	-----
A-017 R 9		A-035 R 8	

A-036 R 7	E		A-037 ÷	-----
<b>REGISTERS</b>				
0	1	2	3	
4	5	6	7	n
8 Used	9 Used	.0	.1	
.2	.3	.4	.5	I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	0.00
3	For expected values unequal, go to step 4; for expected values equal, go to step 8. <b>Unequal</b>			
4	Key in observed and expected value.	$O_i$	ENTER ↴	
		$E_i$	GSB [1]	i
5	Repeat step 4 until all values have been keyed in.			
6	Compute chi-square statistic.		R/S	$\chi^2$
7	For a new case go to step 2. <b>Equal</b>			
8	Key in observed value.	$O_i$	GSB [2]	i
9	Repeat step 8 until all ob- served values have been keyed in.			
10	Compute chi-square statistic.		R/S	$\chi^2$
11	Compute E.		R/S	E
12	For a new case, go to step 2.			

**Example 1:**

Find the  $\chi^2$  statistic for the data below (expected values unequal):

$O_i$	8	50	47	56	5	14
$E_i$	9.6	46.7	51.8	54.4	8.25	9.15

**Keystrokes:**

A	_____	0.00
8	ENTER ↴ 9.6 GSB [1] _____	1.00
50	ENTER ↴ 46.7 GSB [1] _____	2.00
47	ENTER ↴ 51.8 GSB [1] _____	3.00
56	ENTER ↴ 54.4 GSB [1] _____	4.00
5	ENTER ↴ 8.25 GSB [1] _____	5.00
14	ENTER ↴ 9.15 GSB [1] _____	6.00
R/S	_____	4.84 $(\chi^2)$

**Outputs:****Example 2:**

The following table shows the observed frequencies in tossing a die 120 times.  $\chi^2$  can be used to test if the die is fair.

Number	1	2	3	4	5	6
Frequency	25	17	15	23	24	16

**Keystrokes:**

A	_____	0.00
25	GSB [2] _____	1.00
17	GSB [2] _____	2.00
15	GSB [2] _____	3.00
23	GSB [2] _____	4.00
24	GSB [2] _____	5.00
16	GSB [2] _____	6.00
R/S	_____	5.00 $(\chi^2)$
R/S	_____	20.00 $(E)$

**Outputs:**

## PAIRED t STATISTIC

Given a set of paired observations from two normal populations with means  $\mu_1, \mu_2$  (unknown)

$x_i$	$x_1$	$x_2$	$\dots$	$x_n$
$y_i$	$y_1$	$y_2$	$\dots$	$y_n$

let

$$D_i = x_i - y_i$$

$$\bar{D} = \frac{1}{n} \sum_{i=1}^n D_i$$

$$s_D = \sqrt{\frac{\sum D_i^2 - \frac{1}{n} (\sum D_i)^2}{n - 1}}$$

$$s_{\bar{D}} = \frac{s_D}{\sqrt{n}}$$

The test statistic

$$t = \frac{\bar{D}}{s_{\bar{D}}},$$

which has  $n - 1$  degrees of freedom (df), can be used to test the null hypothesis

$$H_0: \mu_1 = \mu_2.$$

$A-000 LBL A$ $A-001 CL \Sigma$ $B-000 LBL B$ $B-001 -$ $B-002 \Sigma +$ $B-003 RTN$ $B-004 LBL 1$ $B-005 S$ $B-006 R.D$		Clear. ----- Sum $x_i - y_i$ . ----- 	$B-007 IX$ $B-008 \div$ $B-009 \bar{x}$ $B-010 LSTX$ $B-011 \div$ $B-012 RTN$ $B-013 R.D$ $B-014 1$ $B-015 -$	$t$ ----- $df$ ----- 
<b>REGISTERS</b>				
0	1	2	3	
4	5	6	7	
8	9	.0 n	.1 $\Sigma D$	
.2 $\Sigma D^2$	.3 Used	.4 Used	.5 Used	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	
3	Key in x- and y-values.	$x_i$	ENTER	
		$y_i$	B	i
4	Repeat step 3 until all values have been keyed in.			
5	Compute t statistic and degrees of freedom.		GSB [1]	t
			R/S	df

**Example:**

Find the paired t-statistic for the data below:

x	14	17.5	17	17.5	15.4
y	17	20.7	21.6	20.9	17.2

**Keystrokes:**

A 14 ENTER B 17 B →  
 17.5 ENTER B 20.7 B →  
 17 ENTER B 21.6 B →  
 17.5 ENTER B 20.9 B →  
 15.4 ENTER B 17.2 B →  
 GSB [1] →  
 R/S →

**Outputs:**

1.00  
 2.00  
 3.00  
 4.00  
 5.00 (n)  
 -7.16 (t)  
 4.00 (df)

## t STATISTIC FOR TWO MEANS

Suppose  $\{x_1, x_2, \dots, x_{n_1}\}$  and  $\{y_1, y_2, \dots, y_{n_2}\}$  are independent random samples from two normal populations having means  $\mu_1, \mu_2$  (unknown) and the same unknown variance  $\sigma^2$ .

We want to test the null hypothesis

$$H_0: \mu_1 - \mu_2 = D$$

where D is a given number.

Define

$$\bar{x} = \frac{1}{n_1} \sum_{i=1}^{n_1} x_i$$

$$\bar{y} = \frac{1}{n_2} \sum_{i=1}^{n_2} y_i$$

$$t = \frac{\bar{x} - \bar{y} - D}{\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \sqrt{\frac{\sum x_i^2 - n_1 \bar{x}^2 + \sum y_i^2 - n_2 \bar{y}^2}{n_1 + n_2 - 2}}$$

We can use this t statistic, which has the t distribution with  $n_1 + n_2 - 2$  degrees of freedom, to test the null hypothesis  $H_0$ .

$A-000 LBLH$ $A-001 CL \Sigma$ $B-000 LBLB$ $B-001 R .0$ $B-002 S .0$ $B-003 R .2$ $B-004 S 1$ $B-005 \bar{x}$ $B-006 S 2$ $B-007 CL \Sigma$ $B-008 RTN$ $B-009 \bar{x}$ $B-010 S 3$ $B-011 LSTX$ $B-012 +$ $B-013 CHS$ $B-014 R 2$ $B-015 +$ $B-016 R 0$ $B-017 1/X$ $B-018 R .0$ $B-019 1/X$	Clear. ----- Save parameters of x-values. ----- Clear $\Sigma$ for y-values. -----	$B-020 +$ $B-021 IX$ $B-022 \div$ $B-023 R 1$ $B-024 R 2$ $B-025 X^2$ $B-026 R .0$ $B-027 x$ $B-028 -$ $B-029 R .2$ $B-030 +$ $B-031 R 3$ $B-032 X^2$ $B-033 R .0$ $B-034 x$ $B-035 -$ $B-036 R .0$ $B-037 R .0$ $B-038 +$ $B-039 .2$ $B-040 -$ $B-041 \div$
---	---	---

B-042 JX		B-043 ÷	t
<b>REGISTERS</b>			
0 $n_1$	1 $\Sigma x^2$	2 $\bar{x}$	3 $\bar{y}$
4	5	6	7
8	9	.0 $n_2$	.1 $\Sigma y$
.2 $\Sigma y^2$	.3 Used	.4 Used	.5 Used I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		A	
3	Key in x-value.	$x_i$	$\Sigma+$	$\bar{x}$
4	Repeat step 4 for all x-values.			
5	Compute mean of x-values.		B	$\bar{x}$
6	Key in y-value.	$y_i$	$\Sigma+$	i
7	Repeat step 6 for all y-values.			
8	Input D and compute t.	D	R/S	t
9	To find the mean of y-values.		$\bar{x}$	$\bar{y}$

**Example:**

Test the hypothesis  $\mu_1 = \mu_2$  (i.e.,  $D = 0$ ) for the data below:

$$x: 79, 84, 108, 114, 120, 103, 122, 120$$

$$y: 91, 103, 90, 113, 108, 87, 100, 80, 99, 54.$$

**Keystrokes:**

**A** 79  $\Sigma+$  84  $\Sigma+$  → 2.00  
 108  $\Sigma+$  114  $\Sigma+$  120  $\Sigma+$  → 5.00  
 103  $\Sigma+$  122  $\Sigma+$  120  $\Sigma+$  → 8.00  
**B** → 106.25  
 91  $\Sigma+$  103  $\Sigma+$  90  $\Sigma+$  → 3.00  
 113  $\Sigma+$  108  $\Sigma+$  87  $\Sigma+$  → 6.00  
 100  $\Sigma+$  80  $\Sigma+$  99  $\Sigma+$   
 54  $\Sigma+$  →

**Outputs:**

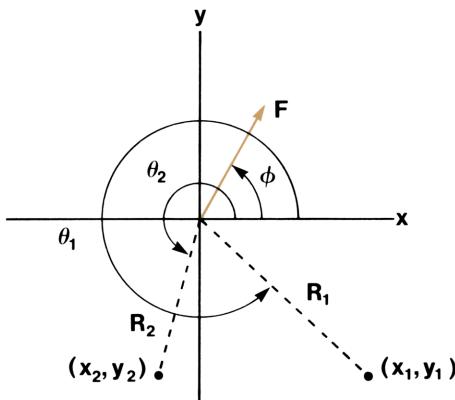
2.00	
5.00	( $n_1$ )
8.00	
106.25	( $\bar{x}$ )
3.00	
6.00	
10.00	( $n_2$ )

0	R/S	→	1.73	(t)
$\bar{x}$		→	92.50	(y)

# STRUCTURES

## STATIC EQUILIBRIUM OF A POINT

This program calculates the two reaction forces necessary to balance a given two-dimensional force vector. The direction of the reaction forces may be specified as a vector of arbitrary length or by Cartesian coordinates using the point of force application as the origin.



### Equations:

$$R_1 \cos \theta_1 + R_2 \cos \theta_2 = F \cos \phi$$

$$R_1 \sin \theta_1 + R_2 \sin \theta_2 = F \sin \phi$$

where:

$F$  is the known force;

$\phi$  is the direction of the known force;

$R_1$  is one reaction force;

$\theta_1$  is the direction of  $R_1$ ;

$R_2$  is the second reaction force;

$\theta_2$  is the direction of  $R_2$ ;

The coordinates  $x_1$  and  $y_1$  are referenced from the point where  $F$  is applied to the end of the member along which  $R_1$  acts;  $x_2$  and  $y_2$  are the coordinates referenced from the point where  $F$  is applied to the end of the member along which  $R_2$  acts.

**Remarks:**

This program assumes the calculator is set in DEG mode. This program may be used in conjunction with the Vector Operations program and leaves registers R<sub>6</sub>–R<sub>9</sub> available for combining the programs.

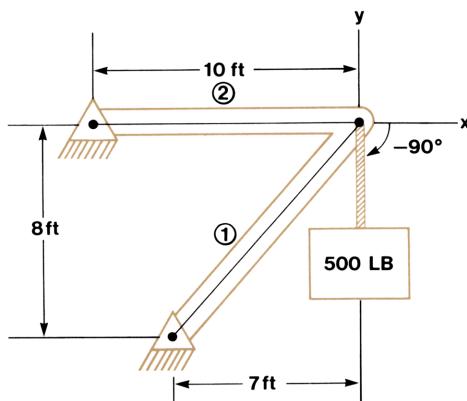
A-000 LBLA		A-027 R 3	
A-001 LBL1	Y <sub>1</sub> , x <sub>1</sub> input	A-028 x	
A-002 →P		A-029 R 5	
A-003 X↔Y		A-030 R 2	
A-004LBL2	-----	A-031 x	
A-005 1	Input θ <sub>1</sub>	A-032 -	
A-006 →R	and store	A-033 R 1	
A-007 S 0		A-034 R 2	
A-008 X↔Y	sin θ <sub>1</sub> , cos θ <sub>1</sub>	A-035 x	
A-009 S 1		A-036 R 0	
A-010 R/S	-----	A-037 R 3	
A-011LBL3	Y <sub>2</sub> , x <sub>2</sub> input	A-038 x	
A-012 →P		A-039 -	
A-013 X↔Y	-----	A-040 ÷	
A-014LBL4		A-041 SPC	
A-015 1	Input θ <sub>2</sub>	A-042 PRTX	
A-016 →R	and store	A-043 LSTX	
A-017 S 2		A-044 R 5	
A-018 X↔Y	sin θ <sub>2</sub> , cos θ <sub>2</sub>	A-045 R 0	
A-019 S 3		A-046 x	
A-020 R/S	-----	A-047 R 4	
A-021LBL5	Input φ and F	A-048 R 1	
A-022 →R		A-049 x	
A-023 S 4	and compute	A-050 -	
A-024 X↔Y		A-051 R↑	
A-025 S 5	reaction forces,	A-052 ÷	
A-026 R 4		A-053 PRTX	
REGISTERS			
0 cos θ <sub>1</sub>	1 sin θ <sub>1</sub>	2 cos θ <sub>2</sub>	3 sin θ <sub>2</sub>
4 F cos φ	5 F sin φ	6	7
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Define reaction directions as Cartesian coordinates or as vectors of arbitrary magnitude. (Use the point of force appli- cations as the origin):			
	define direction one in rec-			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	tangular form	$y_1$	<b>ENTER</b>	$y_1$
		$x_1$	<b>GSB</b> [1]	$\sin \theta_1$
	or in polar form	$\theta_1$	<b>GSB</b> [2]	$\sin \theta_1$
	and			
	define direction two in rec-			
	tangular form	$y_2$	<b>ENTER</b>	$y_2$
	or in polar form	$x_2$	<b>GSB</b> [3]	$\sin \theta_2$
3	Key in known force			
	direction	$\phi$	<b>ENTER</b>	
	then magnitude and compute			
	reactions	$F$	<b>GSB</b> [5]	$R_1, R_2$
4	To change force, go to step 3.			
	To change either or both reac-			
	tion directions, go to step 2.			

**Example 1:**

Find the reaction forces in the pin-jointed structure shown below.



**Keystrokes:**

8 [CHS] [ENTER] 7 [CHS] [GSB] 1 →

0 [ENTER] 10 [CHS] [GSB] 3 →

90 [CHS] [ENTER] 500 [GSB] 5 →

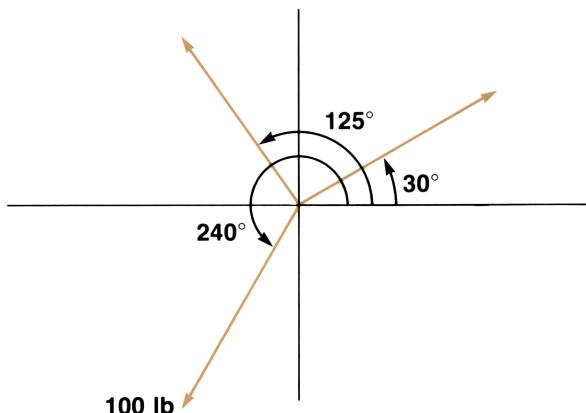
**Outputs:**

-0.75

0.06

-664.38 \*\*\* (R<sub>1</sub>)437.50 \*\*\* (R<sub>2</sub>)**Example 2:**

Find the reaction forces for the diagram below.

**Keystrokes:**

30 [GSB] 2 →

125 [GSB] 4 →

240 [ENTER] 100 [GSB] 5 →

**Outputs:**

0.50

0.82

90.98 \*\*\* (R<sub>1</sub>)50.19 \*\*\* (R<sub>2</sub>)

## COMPOSITE SECTION PROPERTIES

The properties of arbitrarily shaped sections can be evaluated using this program. Exact solutions are obtained when the section is broken into a finite number of rectangles. Approximate solutions can be achieved by assuming that finite areas are concentrated at their centers.

The program calculates the area of the section, the centroid of the area, the moments of inertia about any specified set of axes, the polar moment of inertia about the specified axis, the moments of inertia about an axis translated to the centroid, the moments of inertia of the principal axis, the rotation angle between the translated axis and the principal axis, and the polar moment of inertia about the principal axis.

### **Equations:**

$$A_{si} = \Delta x_i \Delta y_i$$

$$A = A_{s1} + A_{s2} + A_{s3} + \dots + A_{sn}$$

$$\bar{x} = \frac{\sum_{i=1}^n x_{0i} A_{si}}{A}$$

$$\bar{y} = \frac{\sum_{i=1}^n y_{0i} A_{si}}{A}$$

$$I_x = \sum_{i=1}^n \left( y_{0i}^2 + \frac{\Delta y_i^2}{12} \right) A_{si}$$

$$I_y = \sum_{i=1}^n \left( x_{0i}^2 + \frac{\Delta x_i^2}{12} \right) A_{si}$$

$$J = I_x + I_y$$

$$I_{xy} = \sum_{i=1}^n x_{0i} y_{0i} A_{si}$$

$$I_{\bar{x}} = I_x - A \bar{y}^2$$

$$I_{\bar{y}} = I_y - A\bar{x}^2$$

$$I_{\bar{x}\bar{y}} = I_{xy} - A\bar{x}\bar{y}$$

$$\phi = \frac{1}{2} \tan^{-1} \left( \frac{-2I_{\bar{x}\bar{y}}}{I_{\bar{x}} - I_{\bar{y}}} \right)$$

$$I_x' = I_{\bar{x}} \cos^2 \theta + I_{\bar{y}} \sin^2 \theta - I_{\bar{x}\bar{y}} \sin 2\theta$$

$$I_y' = I_{\bar{y}} \cos^2 \theta + I_{\bar{x}} \sin^2 \theta + I_{\bar{x}\bar{y}} \sin 2\theta$$

$$J' = I_x' + I_y'$$

$$I_{xy}' = \frac{(I_{\bar{x}} - I_{\bar{y}})}{2} \sin 2\theta + I_{\bar{x}\bar{y}} \cos 2\theta$$

where:

$\Delta x_i$  is the width of a rectangular element;

$\Delta y_i$  is the height of a rectangular element;

$A_{si}$  is the area of an element;

$A$  is the total area of the section;

$\bar{x}$  is the  $x$  coordinate of the centroid;

$\bar{y}$  is the  $y$  coordinate of the centroid;

$x_{oi}$  is the  $x$  coordinate of the centroid of an element;

$y_{oi}$  is the  $y$  coordinate of the centroid of an element;

$I_x$  is the moment of inertia about the  $x$ -axis;

$I_y$  is the moment of inertia about the  $y$ -axis;

$J$  is the moment of inertia about the origin;

$I_{xy}$  is the product of inertia;

$I_{\bar{x}}$  is the moment of inertia about the  $x$ -axis translated to the centroid;

$I_{\bar{y}}$  is the moment of inertia about the  $y$ -axis translated to the centroid;

$I_{\bar{x}\bar{y}}$  is the product of inertia about the translated axis;

$\phi$  is the angle between the translated axis and the principal axis;

$\theta$  is the angle between the original axis and an arbitrary axis;

$I_x'$  is the  $x$  moment of inertia about the arbitrary axis;

$I_y'$  is the  $y$  moment of inertia about the arbitrary axis;

$J'$  is the polar moment of inertia about the arbitrary axis;

$I_{xy}'$  is the product of inertia about the arbitrary axis.

**Remarks:**

This program assumes the calculator is set in DEG mode.

**Reference:**

Wojciechowski, Felix; "Properties of Plane Cross Sections"; *Machine Design*; P. 105, Jan 22, 1976.

A-000 LBLA		B-002 R 5	
A-001 S 6		B-003 R 0	
A-002 R↓		B-004 R.0	
A-003 S 7	Store $x_0$ , and $y_0$ .	B-005 R.1	
A-004 R/S	-----	B-006 X	
A-005 S 8	Input $\Delta x$ and $\Delta y$ and accumulate moments and area.	B-007 X	
A-006 X $\Rightarrow$ Y	-----	B-008 -	
A-007 X		B-009 ENT↑	
A-008 S.5		B-010 +	
A-009 LSTX		B-011 R 4	
A-010 LBL2		B-012 R.1	
A-011 X $^2$		B-013 X $^2$	
A-012 1		B-014 R 0	
A-013 2		B-015 X	
A-014 ÷		B-016 -	
A-015 R 7		B-017 R.0	
A-016 X $^2$		B-018 X $^2$	
A-017 +		B-019 R 0	
A-018 X		B-020 X	
A-019 S+3		B-021 CHS	
A-020 R 8		B-022 R 3	
A-021 X $^2$		B-023 +	
A-022 1		B-024 -	
A-023 2		B-025 X#0	
A-024 ÷		B-026 ÷	
A-025 R 6		B-027 TAN $^{-1}$	
A-026 X $^2$		B-028 Z	
A-027 +		B-029 ÷	
A-028 R.5		B-030 R 0	
A-029 X		B-031 PRTX	
A-030 S+4		B-032 R.0	
A-031 R.5		B-033 PRTX	
A-032 R 6		B-034 R.1	
A-033 X		B-035 PRTX	
A-034 S+1		B-036 RT↑	
A-035 R 7		B-037 PRTX	
A-036 R.5		B-038 R/S	
A-037 X		B-039 LBL1	
A-038 S+2		B-040 R 1	
A-039 X		B-041 R 0	
A-040 R.5		B-042 ÷	
A-041 ÷		B-043 S.1	
A-042 S+5		B-044 R 2	
A-043 R.5		B-045 R 0	
A-044 S+0		B-046 ÷	
A-045 R/S		B-047 S.0	
A-046 LBL1	Input area for approximate solution.	C-000 LBLC	
A-047 S.5	-----	C-001 ENT↑	
A-048 0	-----	C-002 S 1	
A-049 S 8	-----	C-003 +	
A-050 GT02	Compute $\bar{x}$ and $\bar{y}$ .	C-004 S 9	
B-000 LBLB	-----	C-005 R↓	
B-001 GSB1	-----	C-006 S.3	

C-007 R↓ C-008 S.2 C-009 GSB1 C-010 x C-011 R 0 C-012 x C-013 CHS C-014 R 5 C-015 + C-016 R.3 C-017 R.1 C-018 - C-019 R.2 C-020 R.0 C-021 - C-022 x C-023 R 0 C-024 x C-025 + C-026 S.4 C-027 R.2 C-028 X² C-029 R.2 C-030 R.0 C-031 x C-032 2 C-033 x C-034 - C-035 R 0 C-036 x C-037 R 3 C-038 + C-039 S.2 C-040 R.3 C-041 X² C-042 R.3 C-043 R.1 C-044 x C-045 2 C-046 x C-047 - C-048 R 0 C-049 x C-050 R 4 C-051 + C-052 S.3 C-053 I	Calculate $I_x'$ , $I_y'$ , $J'$ , and $I_{xy}'$ .	C-054 1 C-055 +R C-056 X² C-057 S 6 C-058 R.2 C-059 x C-060 XY C-061 X² C-062 S 7 C-063 R.3 C-064 x C-065 + C-066 R 9 C-067 SIN C-068 R.4 C-069 x C-070 - C-071 PRTX C-072 LSTX C-073 R 6 C-074 R.3 C-075 x C-076 + C-077 R 7 C-078 R.2 C-079 x C-080 + C-081 PRTX C-082 + C-083 PRTX C-084 R.2 C-085 R.3 C-086 - C-087 2 C-088 ÷ C-089 R 9 C-090 SIN C-091 x C-092 R.4 C-093 R 9 C-094 COS C-095 x C-096 + C-097 PRTX C-098 R/S C-099LBL1 C-100JPB1	Output $I_x'$  Output $I_y'$  Output $J'$  Output $I_{xy}'$  Subroutine to B1.
REGISTERS			
0 ΣA	1 $\Sigma x_{0i} A_{si}$	2 $\Sigma y_{0i} A$	3 $\Sigma I_x$
4 $\Sigma I_y$	5 $\Sigma I_{xy}$	6 $x_{0i}, \cos^2 \theta$	7 $y_{0i}, \sin^2 \theta$
8 $\Delta x_{0i}$	9 $2\theta$	.0 $\bar{y}$	.1 $\bar{x}$
.2 y, Used	.3 x, Used	.4 Used	.5 $A_{si}$
			I $\theta$

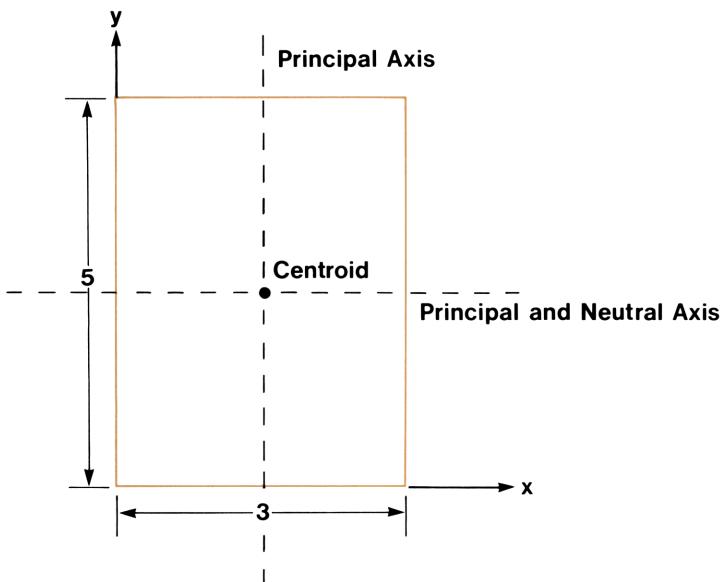
STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		<b>f REG</b>	
3	Key in coordinates of centroid of element.	$y_{oi}$	<b>ENTER</b>	$y_{oi}$
		$x_{oi}$	<b>A</b>	$y_{oi}$
4	Key in height and width of rec- tangular element. <i>or key in area of element for approximate solution.</i>	$\Delta_{yi}$ $\Delta_{xi}$ $A_{si}$	<b>ENTER</b> <b>R/S</b> <b>GSB 1</b>	$\Delta_{yi}$ $A_{si}$ $A_{si}$
5	Repeat steps 3-4 for each element.			
6	Calculate area, centroid, and $\phi$ .		<b>B</b>	$A, \bar{y}, \bar{x}, \phi$
7	Specify arbitrary origin and rotation and calculate properties*	$y$ $x$ $\theta$	<b>ENTER</b> <b>ENTER</b> <b>C</b>	$I_x', I_y', J', I_{xy}'$
	*Note <b>C</b> may be pressed immediately after <b>B</b> for $I_{\bar{x}\phi}$ , $I_{\bar{y}\phi}$ , $J_\phi$ , $I_{\bar{x}\bar{y}\phi}$ since <b>B</b> leaves $\bar{y}, \bar{x}, \phi$ in position in the stack.			
	$I_{\bar{x}}, I_{\bar{y}}, J, I_{\bar{x}\bar{y}}$ may be obtained by <b>B CLX C</b> , or by recalling y and x from storage ( <b>RCL</b> <b>00</b> <b>RCL</b> <b>01</b> <b>0 C</b> ).			

**Example 1:**

What is the moment of inertia about the neutral axis through the centroid of the section ( $I_{\bar{x}\phi}$ ) for the rectangular section shown? What is the moment of inertia about the X-axis ( $I_x$ )?

**Table of Inputs**

Section	$y_0$	$x_0$	$\Delta y$	$\Delta x$
1	2.5	1.5	5	3

**Keystrokes:**

**A** **g REG** 2.5 **ENTER ↴** 1.5 **A**  
**B** **ENTER ↴** 3 **R/S** →

**C** →

(Note the centroid and  $\phi$  are already properly positioned for calculation of properties about the neutral axis through the centroid.)

**Outputs:**

15.00      ( $A_{si}$ )

15.00 \*\*\* ( $A$ )

2.50 \*\*\* ( $\bar{y}$ )

1.50 \*\*\* ( $\bar{x}$ )

0.00 \*\*\* ( $\phi$ )

31.25 \*\*\* ( $I_{\bar{x}\phi}$ )

11.25 \*\*\* ( $I_{\bar{y}\phi}$ )

42.50 \*\*\* ( $J_\phi$ )

0.00 \*\*\* ( $I_{\bar{x}\bar{y}\phi}$ )

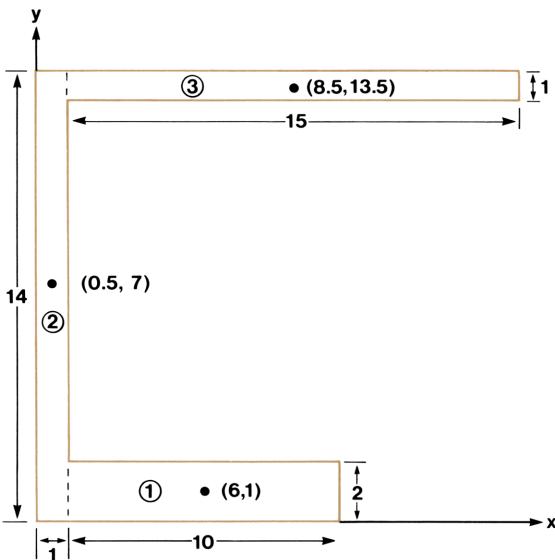
0 [ENTER] 0 [ENTER] 0 [C] →      125.00 \*\*\* ( $I_x$ )  
 (Key in zeroes for properties about      45.00 \*\*\* ( $I_y$ )  
 the origin and axes shown.)      170.00 \*\*\* ( $J$ )  
     56.25 \*\*\* ( $I_{xy}$ )

**Example 2:**

Calculate the section properties for the beam shown below.

**Table of Inputs**

Section	$y_{0i}$	$x_{0i}$	$\Delta v$	$\Delta x$
1	1	6	2	10
2	7	0.5	14	1
3	13.5	8.5	1	15

**Keystrokes:**

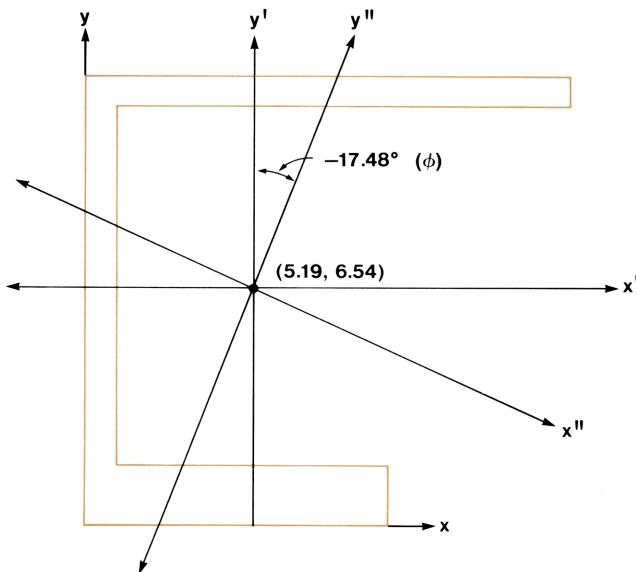
9 [REG] 1 [ENTER] 6 [A] 2 [ENTER]  
 10 [R/S] 7 [ENTER] .5 [A] 14 [ENTER]  
 1 [R/S] 13.5 [ENTER] 8.5 [A]  
 1 [ENTER] 15 [R/S] B →

**Outputs:**

49.00 \*\*\* (A)  
 6.54 \*\*\* ( $\bar{y}$ )

<b>C</b>	→	5.19 *** ( $\bar{x}$ ) -17.48 *** ( $\phi$ )
<b>RCL</b> <b>[C]</b>	→	1651.04 *** ( $I_{\bar{x}\phi}$ ) 863.46 *** ( $I_{\bar{y}\phi}$ ) 2514.49 *** ( $J_\phi$ ) 0.00 *** ( $I_{\bar{x}\bar{y}\phi}$ )
<b>RCL</b> <b>[C]</b>	→	6.54 ( $\bar{y}$ )
<b>RCL</b> <b>[C]</b>	→	5.19 ( $\bar{x}$ )
0		0. ( $\theta$ )
<b>C</b>	→	1580.00 *** ( $I_{\bar{x}}$ ) 934.49 *** ( $I_{\bar{y}}$ ) 2514.49 *** ( $J$ ) 225.61 *** ( $I_{\bar{x}\bar{y}}$ )
0 <b>ENTER</b> 0 <b>ENTER</b> 0 <b>C</b>	→	3676.33 *** ( $I_x$ ) 2256.33 *** ( $I_y$ ) 5932.67 *** ( $J$ ) 1890.25 *** ( $I_{xy}$ )

Below is a figure showing the translated axis and the rotated, principal axis of example 2.

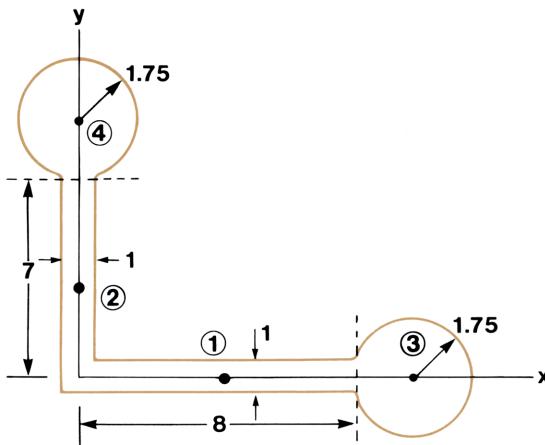


**Example 3:**

Find the approximate section properties of the member below by assuming that the circular parts are concentrated at their centers. Also assume that the boundaries between the circular and rectangular parts are straight lines.

**Table of Inputs**

Section	$y_{0i}$	$x_{0i}$	$\Delta y$	$\Delta x$	A
1	0.0	4.5	1	8	
2	3.0	0.0	7	1	
3	0.0	10.25			9.62
4	8.25	0.0			9.62

**Keystrokes:**

```

g [REG] 0 ENTER 4.5 A 1 ENTER
8 R/S 3 ENTER 0 A 7 ENTER
1 R/S 0 ENTER 10.25 A 9.62
GSB 1 8.25 ENTER 0 A
9.62 GSB 1 B

```

**Outputs:**

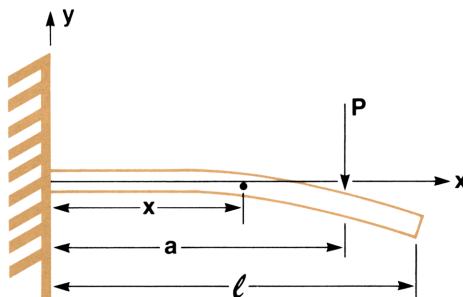
34.24 \*\*\* (A)  
 2.93 \*\*\* ( $\bar{y}$ )  
 3.93 \*\*\* ( $\bar{x}$ )  
 -36.74 \*\*\* ( $\phi$ )

<b>C</b>	→	158.27 *** ( $I_{\bar{x}\phi}$ ) 981.34 *** ( $I_{\bar{y}\phi}$ ) 1139.61 *** ( $J_\phi$ ) 0.00 *** ( $I_{\bar{x}\bar{y}\phi}$ )
<b>RCL</b> [•] <b>0</b>	→	2.93      ( $\bar{y}$ )
<b>RCL</b> [•] <b>1</b>	→	3.93      ( $\bar{x}$ )
0 <b>C</b>	→	452.82 *** ( $I_{\bar{x}}$ ) 686.79 *** ( $I_{\bar{y}}$ ) 1139.61 *** ( $J$ ) -394.56 *** ( $I_{\bar{x}\bar{y}}$ )
0 <b>ENTER</b> 0 <b>ENTER</b> 0 <b>C</b>	→	747.01 *** ( $I_x$ ) 1215.95 *** ( $I_y$ ) 1962.96 *** ( $J$ ) 0.00 *** ( $I_{xy}$ )

## CANTILEVER BEAMS—POINT LOAD

This program calculates deflection, slope, moment and shear at any specified point along a rigidly fixed, cantilever beam of uniform cross section for point loads. By using the principle of superposition, complicated beams with multiple point loads may be analyzed.

### Equations:



$$y = \frac{PX_1^2}{6EI} (X_1 - 3a) - \frac{Pa^2}{2EI} (x - a) (x > a)^* \quad (\text{deflection due to point load})$$

$$\theta = \frac{PX_1}{2EI} (X_1 - 2a) \quad (\text{slope due to point load})$$

$$M = P(X_1 - a) \quad (\text{moment due to point load})$$

$$V = P(x \leq a) \quad (\text{shear due to point load})$$

where

$y$  is the deflection at a distance  $x$  from the wall;

$\theta$  is the slope (change in  $y$  per change in  $x$ ) at  $x$ ;

$M$  is the moment at  $x$ ;

$V$  is the shear at  $x$ ;

$I$  is the moment of inertia of the beam;

$E$  is the modulus of elasticity of the beam;

$\ell$  is the length of the beam;

$P$  is a concentrated load;

$a$  is the distance from the foundation to the point load.

\*The notation  $(x > a)$  is interpreted as 1.00 if  $x$  is greater than  $a$  and as 0.00 if  $x$  is less than or equal to  $a$ .  $x_1 = x$  for  $x$  less than  $a$ ;  $x_1 = a$  for  $x$  greater than  $a$ .

### Remarks:

Deflections must not significantly alter the geometry of the problem. Beams must be of constant cross section for deflection and slope equations to be valid. Stresses must be in the elastic region.

Registers  $R_{.0}$ – $R_{.5}$  are available for user storage.

Sums of  $y$ ,  $\theta$ ,  $M$  and  $V$  may be stored in  $R_{.1}$ ,  $R_{.2}$ ,  $R_{.3}$  and  $R_{.4}$ , respectively.

Refer to page 128 for sign conventions for beams.

A-000 LB <sub>4</sub>		A-031 R 4	
A-001 S 0	Store specified point	A-032 -	
A-002 S 0		A-033 X<0	
A-003 R 4		A-034 CL X	
A-004 X<Y		A-035 x	
A-005 S E		A-036 -	
A-006 R E		A-037 R/S	
A-007 R 4		A-038 R 6	
A-008 5		A-039 R 4	
A-009 x		A-040 +	
A-010 -		A-041 S 6	
A-011 S 6		A-042 R 7	
A-012 R 5		A-043 x	Compute θ.
A-013 R 2		A-044 R 8	
A-014 ÷		A-045 x	
A-015 R 1		A-046 R/S	
A-016 ÷		A-047 R 5	
A-017 2		A-048 R 6	Compute M.
A-018 ÷		A-049 R 4	
A-019 S 7		A-050 +	
A-020 5		A-051 x	
A-021 ÷	Compute y.	A-052 R/S	
A-022 x		A-053 R 5	
A-023 R 8		A-054 R 4	
A-024 X <sup>2</sup>		A-055 R 0	
A-025 x		A-056 -	Compute V.
A-026 R 4		A-057 X<0	
A-027 X <sup>2</sup>		A-058 CL X	
A-028 R 7		A-059 X<0	
A-029 x		A-060 X <sup>2</sup> Y	
A-030 R 0			

## REGISTERS

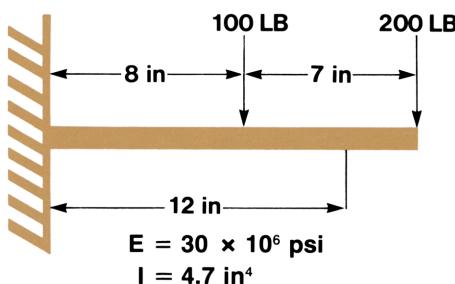
0 x	1 I	2 E	3 ℓ
4 a	5 P	6 Used	7 <u>P</u> 2EI
8 x, a	9	.0	.1
.2	.3	.4	.5

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store beam parameters:			
	Moment of inertia of beam	I	STO [1]	I
	Modulus of elasticity of beam	E	STO [2]	E
	Length of beam	ℓ	STO [3]	ℓ
	Distance from foundation			
	to point load	a	STO [4]	a
	Concentrated load	P	STO [5]	P

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
3	Key in $x$ to specify point of interest and calculate deflection; calculate slope at $x$ ; calculate moment at $x$ ; calculate shear at $x$ .	$x$	A	y
			R/S	$\theta$
			R/S	M
			R/S	V
4	For a new location with the same loading, go to step 3. For new beam properties or loading, go to step 2 and change the appropriate values.			

**Example 1:**

What is the deflection at  $x = 12$ ? Neglect the weight of the beam.

**Keystrokes:**

4.7 **STO** [1] 30 **EEX** 6 **STO** [2]  
 15 **STO** [3] f **ENG** 3

**Outputs:**

Compute deflection at 12 inches due to 100 lb. weight:

8 **STO** [4] 100 **STO** [5] 12 **A** → -211.8 -06

Store deflection due to 100 lb. load for addition to deflection due to 200 lb. load:

**STO** [6] → -211.8 -06

Compute deflection at 12 inches due to 200 lb. load:

15 **STO** **4** 200 **STO** **5** 12 **A** → -1.123 -03

Compute total deflection.

**RCL** **[ ]** **1** **+** → -1.335 -03

What are the deflection, slope, moment and shear at 15 inches due to the 200 lb. load?

15 **A** → -1.596 -03

**R/S** → -159.6 -06

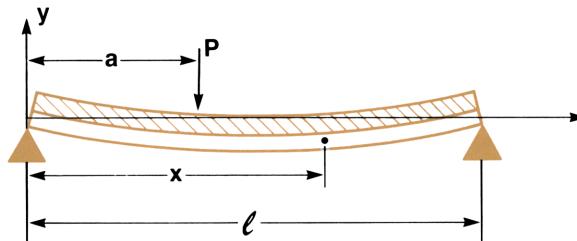
**R/S** → 0.000 00

**R/S** → 0.000 00

## SIMPLY SUPPORTED BEAMS—POINT LOAD

This program calculates deflection, slope, moment and shear at any specified point along a simply supported beam of uniform cross section for point loads. By using the principle of superposition, complicated beams with multiple point loads can be analyzed.

**Equations:**



$$y = \frac{P(\ell-a)x}{6EI} [x^2 + (\ell-a)^2 - \ell^2]^* \quad (\text{deflection due to point load})$$

$$\theta = \frac{P(\ell-a)}{6EI} [3x^2 + (\ell-a)^2 - \ell^2]^* \quad (\text{slope due to point load})$$

$$M = \frac{P(\ell-a)x}{\ell}^* \quad (\text{moment due to point load})$$

$$V = \frac{P(\ell-a)}{\ell}^* \quad (\text{shear due to point load})$$

\*If  $x$  is greater than  $a$ ,  $(\ell - a)$  is replaced by  $-a$  and  $x$  is replaced by  $x - \ell$ .

where

$y$  is the deflection at a distance  $x$  from the left support;

$\theta$  is the slope (change in  $y$  per change in  $x$ ) at  $x$ ;

$M$  is the moment at  $x$ ;

$V$  is the shear at  $x$ ;

$I$  is the moment of inertia of the beam;

$E$  is the modulus of elasticity of the beam;

$\ell$  is the length of the beam;

$P$  is the length of the beam;

$a$  is the distance from the left support to the point load.

### Remarks:

Deflections must not significantly alter the geometry of the problem. Beams must be of constant cross section for deflection and slope equations to be valid. Stresses must be in the elastic region.

Registers R.<sub>0</sub>–R.<sub>5</sub> are available for user storage.

Sums of  $y$ ,  $\theta$ ,  $M$  and  $V$  may be stored in R.<sub>1</sub>, R.<sub>2</sub>, R.<sub>3</sub>, and R.<sub>4</sub>, respectively.

Refer to page 128 for sign conventions for beams.

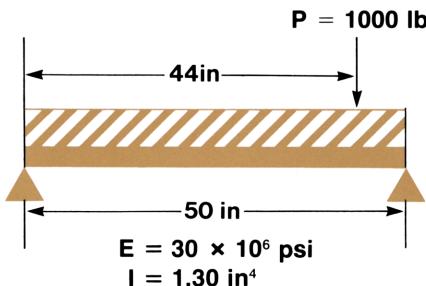
B-000 LBL5		B-026 R/S	
B-001 S 0	Store specified point.	B-027 R 7	
B-002 R 4	-----	B-028 R 0	
B-003 X>Y	Compare x and a.	B-029 x	
B-004 GT01	-----	B-030 R/S	
B-005 CHS		B-031 R 7	
B-006 S 6		B-032 R/S	
B-007 CL X	Store -a and x- $\ell$	B-033 LBL2	
B-008 R 3	-----	B-034 R 0	
B-009 -		B-035 x	
B-010 S 0		B-036 R 6	
B-011 GT01	-----	B-037 +P	
B-012 LBL0		B-038 X <sup>2</sup>	
B-013 R 3		B-039 R 3	
B-014 -		B-040 X <sup>2</sup>	
B-015 CHS	Store $\ell$ -a	B-041 -	
B-016 S 6	-----	B-042 R 5	
B-017 LBL1		B-043 R 6	
B-018 1		B-044 x	
B-019 GSB2	Compute y.	B-045 R 3	
B-020 R 0	-----	B-046 ÷	
B-021 x		B-047 S 7	
B-022 R/S	-----	B-048 R 2	
B-023 3		B-049 ÷	
B-024 FX	Compute $\theta$ .	B-050 R 1	
B-025 GSB2		B-051 ÷	

<b>B-052</b>	<b>6</b>		<b>B-054</b>	<b>x</b>	
<b>B-053</b>	<b>÷</b>		<b>B-055</b>	<b>RTN</b>	
<b>REGISTERS</b>					
0    x, x-l	1    I	2    E	3    l		
4    a	5    P	6    l-a, -a	7 $\frac{P(l-a)}{l}$		
8	9	.0	.1		

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store beam parameters:			
	Moment of inertia of beam	I	STO [1]	I
	Modulus of elasticity of beam	E	STO [2]	E
	Length of beam	l	STO [3]	l
	Distance from foundation			
	to point load	a	STO [4]	a
	Concentrated load	P	STO [5]	P
3	Key in x to specify point of interest and calculate deflection;	x	B	y
	calculate slope at x;		R S	$\theta$
	calculate moment at x;		R S	M
	calculate shear at x.		R S	V
4	For a new location with the same loading, go to step 3.			
	For new beam properties or loading, go to step 2 and change the appropriate values.			

**Example 1:**

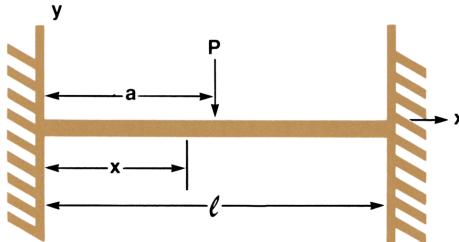
What are the deflection, slope, moment, and shear of the beam below at  $x = 38$  inches?

**Keystrokes:**

```
1.3 [STO] 1 30 [EE] 6 [STO] 2 50
[STO] 3 44 [STO] 4 1000 [STO] 5 [f]
[ENG] 3
38 [B] → -19.88 -03
[R/S] → 957.9 -06
[R/S] → 4.560 03
[R/S] → 120.0 00
```

**Outputs:****FIXED-FIXED BEAM-POINT LOAD**

This program calculates deflection, slope, moment and shear at any specified point along a beam of uniform cross section fixed at both ends for point loads. By using the principle of superposition, complicated beams with multiple point loads can be analyzed.

**Equations:**

$$y = \frac{P(\ell-a)^2 x^2}{6EI\ell^3} [x(\ell+2a) - 3a\ell]^* \quad (\text{deflection due to point load})$$

$$\theta = \frac{P(\ell-a)^2 x}{2EI\ell^3} [x(\ell+2a) - 2a\ell]^* \quad (\text{slope due to point load})$$

$$M = \frac{P(\ell-a)^2}{\ell^3} [x(\ell+2a) - a\ell]^* \quad (\text{moment due to point load})$$

$$V = \frac{P(\ell-a)^2}{\ell^3} (\ell+2a)^* \quad (\text{shear due to point load})$$

\*If  $x$  is greater than  $a$ ,  $a$  is replaced by  $(\ell - a)$  and  $x$  is replaced by  $(\ell - x)$ . The signs of  $\theta$  and  $V$  are also changed.

where

$y$  is the deflection at a distance  $x$  from the left support;

$\theta$  is the slope (change in  $y$  per change in  $x$ ) at  $x$ ;

$M$  is the moment at  $x$ ;

$V$  is the shear at  $x$ ;

$I$  is the moment of inertia of the beam;

$E$  is the modulus of elasticity of the beam;

$\ell$  is the length of the beam;

$P$  is a concentrated load;

$a$  is the distance from the left support to the point load.

### Remarks:

Deflections must not significantly alter the geometry of the problem. Beams must be of constant cross section for deflection and slope equations to be valid. Stresses must be in the elastic region.

Registers  $R_{.1}$ – $R_{.5}$  are available for user storage.

Sums of  $y$ ,  $\theta$ ,  $M$  and  $V$  may be stored in  $R_{.1}$ ,  $R_{.2}$ ,  $R_{.3}$ , and  $R_{.4}$ , respectively.

**Sign Conventions for Beams**

NAME	VARIABLE	SENSE	SIGN
DEFLECTION	y	↑	+
SLOPE	θ	↑	+
INTERNAL MOMENT	M	(—)	+
SHEAR	V	↑—↓	+
EXTERNAL FORCE OR LOAD	P	↓	+

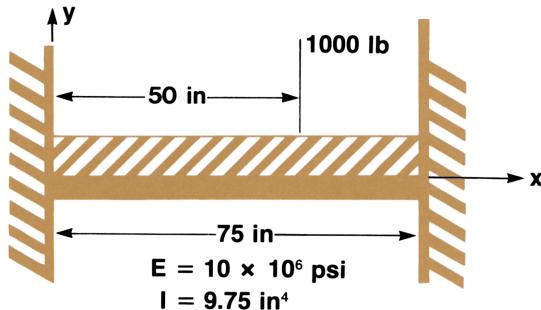
C-000 LBL1	Store specified point.	C-040 ÷	
C-001 S 0	-----	C-041 6	
C-002 R 4	-----	C-042 ÷	
C-003 X>Y	Compare x and a.	C-043 3	
C-004 GTO0	-----	C-044 GSB2	
C-005 R 3	-----	C-045 R/S	
C-006 -	-----	C-046 R 7	
C-007 CHS	-----	C-047 R 0	
C-008 S 6	Store l-a	C-048 x	
C-009 R 3	and l-x	C-049 R.0	
C-010 R 0	-----	C-050 ÷	
C-011 -	-----	C-051 2	
C-012 S 0	-----	C-052 ÷	
C-013 1	-----	C-053 2	
C-014 CHS	-----	C-054 GSB2	
C-015 S 9	-----	C-055 R 9	
C-016 GTO1	-----	C-056 x	
C-017 LBL0	-----	C-057 R/S	
C-018 S 6	Store a.	C-058 R 7	
C-019 1	-----	C-059 1	
C-020 S 9	-----	C-060 GSB2	
C-021 LBL1	-----	C-061 R/S	
C-022 R 3	-----	C-062 R 7	
C-023 R 6	-----	C-063 R 8	
C-024 -	-----	C-064 x	
C-025 X²	-----	C-065 R 9	
C-026 R 5	-----	C-066 x	
C-027 x	$\frac{P(l-a)^2}{l^3}$	C-067 R/S	
C-028 R 3	-----	C-068 LBL2	
C-029 3	-----	C-069 CHS	
C-030 Y'	-----	C-070 R 6	
C-031 ÷	-----	C-071 R 3	
C-032 S 7	-----	C-072 x	
C-033 R 0	-----	C-073 x	
C-034 X²	-----	C-074 R 6	
C-035 x	-----	C-075 2	
C-036 R 1	-----	C-076 x	
C-037 R 2	-----	C-077 R 3	
C-038 x	-----	C-078 +	
C-039 S.0	-----	C-079 S 8	
	EI → R.0		Compute subroutine.

C-080 R 0		C-083 x	
C-081 x		C-084 RTN	
C-082 +			
<b>REGISTERS</b>			
0 x, l-x	1 I	2 E	3 l
4 a	5 P	6 a, l-a	7 $P(l-a)^2/l^3$
8 l+2a	9 ±1	.0 EI	.1
.2	.3	.4	.5
			I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Store beam parameters:			
	Moment of inertia of beam	I	STO [1]	I
	Modulus of elasticity of beam	E	STO [2]	E
	Length of beam	l	STO [3]	l
	Distance from foundation			
	to point load	a	STO [4]	a
	Concentrated load	P	STO [5]	P
3	Key in x to specify point of interest and calculate deflection;	x	C	y
	calculate slope at x;		R/S	θ
	calculate moment at x;		R/S	M
	calculate shear at x.		R/S	V
4	For a new location with the same loading, go to step 3.			
	For new beam properties or loading, go to step 2 and change the appropriate values.			

**Example 1:**

Find the deflection, slope, moment and shear at  $x = 0$  and at  $x = 40$  inches for the configuration below.

**Keystrokes:**

9.75 **STO** **1** 10 **EEX** 6 **STO** **2**  
 75 **STO** **3** 50 **STO** **4** 1000 **STO** **5**

**f** **ENG** 3

0	<b>C</b>	→	0.000	00
	<b>R/S</b>	→	0.000	00
	<b>R/S</b>	→	-5.556	03
	<b>R/S</b>	→	259.3	00
40	<b>C</b>	→	-17.22	-03
	<b>R/S</b>	→	-151.9	-06
	<b>R/S</b>	→	4.815	03
	<b>R/S</b>	→	259.3	00

**Outputs:**

## **NOTES**

# SURVEYING

## FIELD ANGLE TRAVERSE

This program uses horizontal distances and angles or deflections turned from a reference azimuth to compute the coordinates of successive points in a traverse. For a closed traverse, the area enclosed and closure distance and azimuth are computed.

### Equations:

$$N_{i+1} = N_i + HD \cos Az$$

$$E_{i+1} = E_i + HD \sin Az$$

$$\text{Area} = \sum_{k=1}^n LAT_k \left( \frac{1}{2} DEP_k + \sum_{j=1}^{k-1} DEP_j \right)$$

where  $DEP_k = E_{k+1} - E_k$  and  $LAT_k = N_{k+1} - N_k$

### Remarks:

Some programming savings may be made by the user if any of programs Field Angle Traverse, Bearing Traverse, and Inverse are combined.

This program assumes the calculator is set in DEG mode.

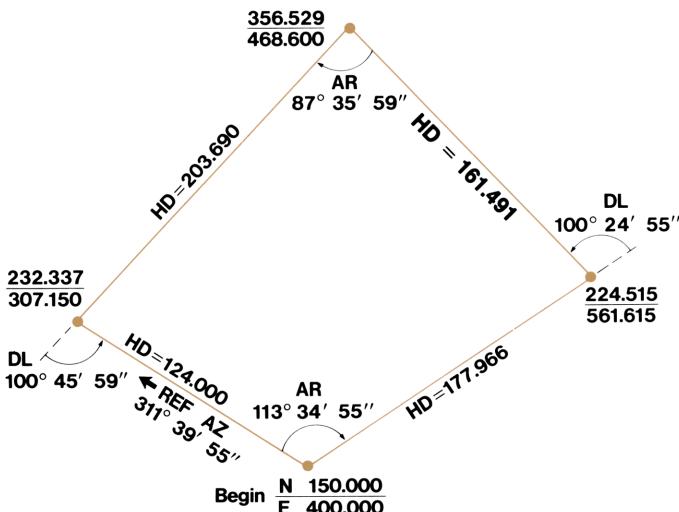
A-000 LBLA		A-025 +P	
A-001 FIX4		A-026 LBL5	
A-002 CL R		A-027 X#Y	
A-003 S 0		A-028 X>0	
A-004 X#Y		A-029 GT03	
N-005 S 1		A-030 3	
A-006 1		A-031 6	
A-007 6		A-032 0	
A-008 0		A-033 +	
A-009 S 2		A-034 LBL3	
A-010 R/S		A-035 S 3	
A-011 R 2		A-036 +HMS	
A-012 HMS+		A-037 R/S	
A-013 +H		A-038 S+4	
A-014 GT04		A-039 R 3	
A-015 LBL1		A-040 X#Y	
A-016 R 2		A-041 +R	
A-017 HNS+		A-042 S+5	
A-018 LBL2		A-043 X#Y	
A-019 +H		A-044 S+6	
A-020 R 3		A-045 2	
A-021 +		A-046 ÷	
A-022 LBL4		A-047 R 6	
A-023 1		A-048 -	
A-024 +R		A-049 x	Compute next coordinates. and accumulate area.

A-050 S+7 A-051 R 5 A-052 R 1 A-053 + A-054 SPC A-055 PRTX A-056 R 6 A-057 R 0 A-058 + A-059 PRTX		D-000 LBLD D-001 R 7 D-002 ABS D-003 R/S D-004 R 6 D-005 R 5 D-006 +P D-007 R/S D-008 JPA5	Area ----- Set-up for closure. -----
<b>REGISTERS</b>			
0 Beg E	1 Beg N	2 180	3 Az
4 ΣHD	5 LAT	6 DEP	7 Area
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in beginning coordinates.	BEG N	ENTER	
		BEG E	A	180.00
3	Key in reference azimuth			
	away from beginning point.	REF AZ(D.MS)	R S	Az (D.MS)
4	Key in field angle:			
	angle right		GSB 1	Az (D.MS)
	or angle left		CHS GSB 1	Az (D.MS)
	or deflection right		GSB 2	Az (D.MS)
	or deflection left		CHS GSB 2	Az (D.MS)
5	Key in horizontal distance and			
	compute coordinates.	HD	R S	N, E
6	Repeat steps 4 and 5 for suc-			
	cessive courses.			
7	For closed figure,			
	compute area,		D	Area
	error distance, and		R S	Error Dist.
	error azimuth		R S	Error Az (D.MS)

**Example:**

Traverse the figure below starting at  $\frac{N \ 150}{E \ 400}$ .

**Keystrokes:**150 **ENTER** 400 **A** →311.3955 **R/S** →113.3455 **GSB** [1] →177.966 **R/S** →100.2455 **CHS** **GSB** [2] →161.491 **R/S** →87.3559 **GSB** [1] →203.690 **R/S** →100.4559 **CHS** **GSB** [2] →124 **R/S** →**D** →**R/S** →**R/S** →**Outputs:**

180.0000

131.3955

65.1450

224.5150 \*\*\* N

561.6150 \*\*\* E

324.4955

356.5285 \*\*\* N

468.6000 \*\*\* E

232.2554

232.3372 \*\*\* N

307.1498 \*\*\* E

131.3955

149.9048 \*\*\* N

399.7829 \*\*\* E

26558.8204 Area

0.2371 Error distance

246.1844 Error azimuth

## BEARING TRAVERSE

This program uses quadrant bearings and horizontal distances to compute the coordinates of successive points in a traverse. For a closed traverse, the area enclosed and error distance and azimuth are computed.

### Equations:

$$N_{i+1} = N_i + HD \cos Az$$

$$E_{i+1} = E_i + HD \sin Az$$

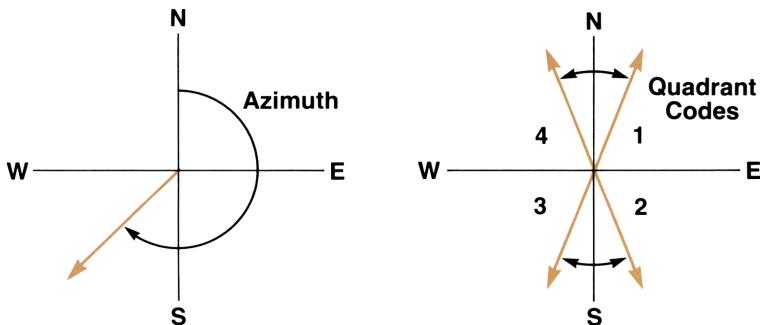
$$\text{Area} = \sum_{k=1}^n \text{LAT}_k \left( \frac{1}{2} \text{DEP}_k + \sum_{j=1}^{k-1} \text{DEP}_j \right)$$

where  $\text{DEP}_k = E_{k+1} - E_k$  and  $\text{LAT}_k = N_{k+1} - N_k$

### Remarks:

Some programming savings may be made by the user if any of programs Field Angle Traverse, Bearing Traverse, and Inverse are combined.

This program assumes the calculator is set in DEG mode.



B-000 LBL E		B-009 S 2	
B-001 FIX4		B-010 R/S	
B-002 CL R		B-011 LBL I	
B-003 S 0		B-012 ENT↑	
B-004 X#Y		B-013 ENT↑	
B-005 S 1	Store starting coordinates and 180.	B-014 2	
B-006 1		B-015 ÷	
B-007 6		B-016 INT	
B-008 6		B-017 R 2	
			Convert bearing and quadrant code to azimuth.

B-018 X		B-046 +R	
B-019 X $\neq$ Y		B-047 S+5	
B-020 R 2		B-048 X $\neq$ Y	
B-021 X		B-049 S+6	
B-022 COS		B-050 Z	
B-023 R↑		B-051 ÷	
B-024 +H		B-052 R 6	
B-025 X		B-053 -	
B-026 -		B-054 X	
B-027 LBL4		B-055 S+7	
B-028 i		B-056 R 5	
B-029 +R		B-057 R 1	
B-030 +P		B-058 +	
B-031 LBL2		B-059 SPC	
B-032 X $\neq$ Y		B-060 PRTX	
B-033 X>0	0 < Az ≤ 360.	B-061 R 6	
B-034 GTO3		B-062 R 0	
B-035 3		B-063 +	
B-036 6		B-064 PRTX	
B-037 6		D-000 LBLD	
B-038 +		D-001 R 7	
B-039 LBL3		D-002 ABS	
B-040 S 3		D-003 R/S	
B-041 +HMS		D-004 R 6	
B-042 R/S		D-005 R 5	
B-043 S+4		D-006 +P	
B-044 R 3		D-007 R/S	
B-045 X $\neq$ Y	Input horizontal distance.	D-008 JPBZ	

## REGISTERS

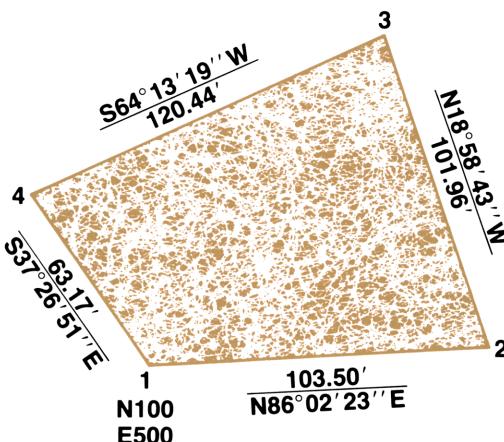
0 Beg E	1 Beg N	2 180	3 Az
4 ΣHD	5 LAT	6 DEP	7 Area
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in beginning coordinates.	BEG N	ENTER	
		BEG E	B	180.00
3	Key in bearing and quadrant code.	BRG (D.MS)	ENTER	
		QD	GSB 1	Az (D.MS)
4	Key in horizontal distance and compute coordinates.	HD	R.S	N, E
5	Repeat steps 3 and 4 for successive courses.			
6	For closed figure, compute area,		D	Area

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	error distance, and		R/S	Error Dist.
	error azimuth		R/S	Error Az (D.MS)

**Example:**

Traverse the figure below starting at  $\frac{\text{N } 100}{\text{E } 500}$ .

**Keystrokes:**

- 100 **ENTER** 500 **B** →
- 86.0223 **ENTER** 1 **GSB** **1** →
- 103.50 **R/S** →
- 18.5843 **ENTER** 4 **GSB** **1** →
- 101.96 **R/S** →
- 64.1319 **ENTER** 3 **GSB** **1** →
- 120.44 **R/S** →
- 37.2651 **ENTER** 2 **GSB** **1** →

**Outputs:**

- 180.0000
- 86.0223
- 107.1482 \*\*\* N  
603.2529 \*\*\* E
- 341.0117
- 203.5657 \*\*\* N  
570.0939 \*\*\* E
- 244.1319
- 151.1880 \*\*\* N  
461.6395 \*\*\* E
- 142.3309

63.17 R/S	→	101.0366 *** N
		500.0490 *** E
D	→	8855.4922 Area
R/S	→	1.0378 Error distance
R/S	→	2.4219 Error azimuth

## INVERSE

This program calculates the distance and azimuth of the line joining two points, given the coordinates of the points. For a closed inverse, the area enclosed and closure distance and azimuth are computed.

### Equations:

$$HD = \sqrt{(N_i - N_{i-1})^2 + (E_i - E_{i-1})^2}$$

$$Az = \tan^{-1} \frac{E_i - E_{i-1}}{N_i - N_{i-1}}$$

$$\text{Area} = \sum_{k=1}^n \text{LAT}_k \left( \frac{1}{2} \text{DEP}_k + \sum_{j=1}^{k-1} \text{DEP}_j \right)$$

where  $\text{DEP}_k = E_{k+1} - E_k$  and  $\text{LAT}_k = N_{k+1} - N_k$

### Remarks:

Some programming savings may be made by the user if any of programs Field Angle Traverse, Bearing Traverse, and Inverse are combined.

This program assumes the calculator is set in DEG mode.

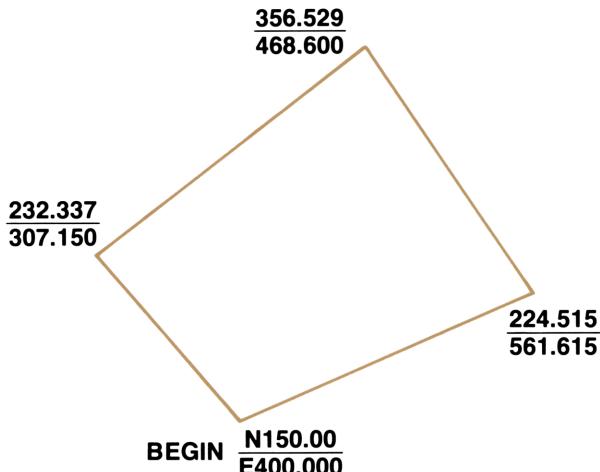
C-000 LBL C		C-016 S+6	
C-001 FIX4		C-017 S 8	
C-002 CL R		C-018 X#Y	
C-003 S 0		C-019 R 5	
C-004 X#Y		C-020 -	
C-005 S 1		C-021 R 1	
C-006 1		C-022 -	
C-007 8		C-023 S+5	
C-008 0		C-024 S 9	
C-009 S 2		C-025 X#Y	
C-010 R/S		C-026 2	
C-011 LBL1		C-027 ÷	
C-012 R 6		C-028 R 6	
C-013 -		C-029 -	
C-014 R 0		C-030 x	
C-015 -		C-031 S+7	

C-032 R 8 C-033 R 9 C-034 LBL2 C-035 →P C-036 SPC C-037 PRTX C-038 X <sub>2</sub> Y C-039 X>0 C-040 GT03 C-041 3 C-042 6 C-043 0	Compute distance and azimuth.	C-044 + C-045 LBL3 C-046 →HMS C-047 PRTX D-000 LBLD D-001 R 7 D-002 ABS D-003 SPC D-004 PRTX D-005 R 6 D-006 R 5 D-007 JPC2	----- Area ----- Set-up for closure. -----
<b>REGISTERS</b>			
0 Beg E	1 Beg N	2 180	3
4	5 LAT	6 DEP	7 Area
8 ΔE	9 ΔN	.0	.1
.2	.3	.4	.5 I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in beginning coordinates.	BEG N	<b>ENTER</b>	
		BEG E	<b>C</b>	180.00
3	Key in coordinates of next point and compute horizontal distance and azimuth.	N E	<b>ENTER</b> <b>GSB [1]</b>	HD Az (D.MS)
4	Repeat step 3 for successive courses.			
6	For closed figure, compute area, error distance, and error azimuth		<b>D</b>	Area Error Dist. Error Az (D.MS)

**Example:**

Inverse the figure below starting at  $\frac{\text{N } 150.00}{\text{E } 400.00}$ .

**Keystrokes:**

150 [ENTER] 400 C →

**Outputs:**

180.0000

224.515 [ENTER] 561.615

177.9660 \*\*\* HD  
65.1450 \*\*\* Az

356.529 [ENTER] 468.600

161.4914 \*\*\* HD  
324.4955 \*\*\* Az

232.337 [ENTER] 307.150

203.6903 \*\*\* HD  
232.2553 \*\*\* Az

150 [ENTER] 400 GSB 1 →

124.0988 \*\*\* HD  
131.3357 \*\*\* Az

D →

26545.4956 \*\*\* Area  
0.0000 \*\*\* Error distance  
360.0000 \*\*\* Error azimuth

Note the area differs slightly from the Field Angle Traverse example area since the area closed exactly for the Inverse.

## SIDESHOTS

This program is used to make sideshots or radials from a point. Two methods may be used for a sideshot, 1) input a bearing and distance and calculate the point coordinates, or 2) input the point coordinates and calculate the azimuth and distance to the point.

### Equations:

$$N = N_0 + HD \cos Az$$

$$E = E_0 + HD \sin Az$$

### Remarks:

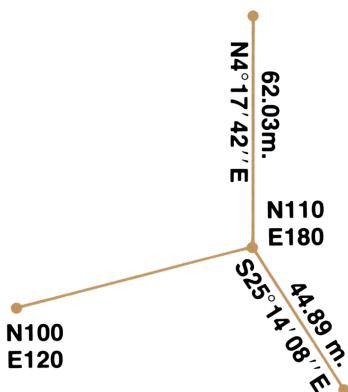
This program assumes the calculator is set in DEG mode.

D-000 LBL0		D-033 X>0	AZ positive?
D-001 FIX4	Store hub coordinates and 180.	D-034 GT03	
D-002 CL R		D-035 3	
D-003 S 0		D-036 6	
D-004 X>Y		D-037 0	
D-005 S 1		D-038 +	
D-006 1		D-039 LBL3	360 + AZ
D-007 8		D-040 S 3	
D-008 0		D-041 +HMS	
D-009 S 2		D-042 RTN	Display AZ
D-010 R/S		D-043 R 3	
D-011 LBL1		D-044 X>Y	
D-012 ENT↑		D-045 +R	
D-013 ENT↑		D-046 R 1	
D-014 2		D-047 +	
D-015 ÷		D-048 SPC	
D-016 INT		D-049 PRTX	
D-017 R 2		D-050 X>Y	Print Northing
D-018 x		D-051 R 0	
D-019 X>Y		D-052 +	
D-020 R 2		D-053 PRTX	Print Easting
D-021 x		D-054 R/S	
D-022 COS		D-055 LBL2	
D-023 R↑		D-056 R 0	
D-024 +H		D-057 -	ΔE
D-025 x		D-058 X>Y	
D-026 -		D-059 R 1	
D-027 LBL4		D-060 -	ΔN
D-028 1		D-061 +P	
D-029 +R		D-062 SPC	
D-030 +P		D-063 PRTX	Print distance
D-031 LBL5		D-064 GS85	Print AZ
D-032 X>Y		D-065 PRTX	
<b>REGISTERS</b>			
0 E <sub>0</sub>	1 N <sub>0</sub>	2 180	3
4	5	6	7

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in hub coordinates	N E	<b>ENTER</b> <b>D</b>	180.00
3a	For bearing sideshot:  Key in bearing and quadrant.  Key in horizontal distance and compute point coordinates.	BRG (D.MS) QD HD	<b>ENTER</b> <b>GSB [1]</b> <b>R/S</b>	Az (D.MS) HD, N, E
3b	For inverse sideshot:  Key in coordinates of sideshot point and compute horizontal distance and azimuth.	N E	<b>ENTER</b> <b>GSB [2]</b>	HD, Az (D.MS)
4	Repeat step 3 or all desired sideshots.			

**Example:**

Calculate the sideshots for the figure shown below:



**Keystrokes:**

110 **ENTER** 180 **D** →  
4.1742 **ENTER** 1 **GSB** **1** →  
62.03 **R/S** →  
25.1408 **ENTER** 2 **GSB** **1** →  
44.89 **R/S** →  
100 **ENTER** 120 **GSB** **2** →

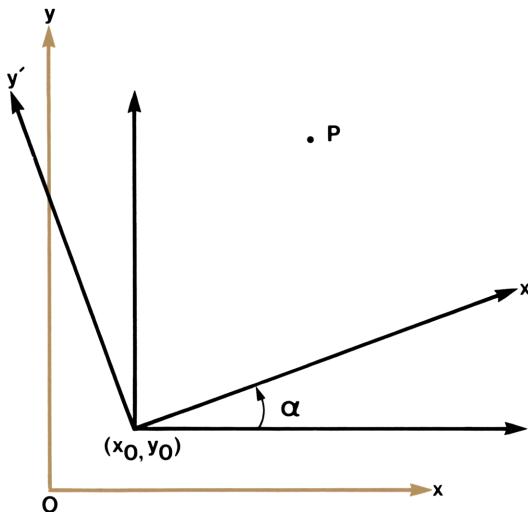
**Outputs:**

180.0000  
4.1742 Az  
171.8558 \*\*\* N  
184.6455 \*\*\* E  
154.4552 Az  
69.3942 \*\*\* N  
199.1384 \*\*\* E  
60.8276 \*\*\* HD  
260.3216 \*\*\* Az

# TRIGONOMETRY AND ANALYTICAL GEOMETRY

## COORDINATE TRANSLATION AND ROTATION

This program allows for two-dimensional translation and rotation of coordinate axes. Suppose the origin of a coordinate system is translated to a new point,  $(x_0, y_0)$ , and the  $x$  and  $y$  axes are rotated through an angle  $\alpha$  to give new axes,  $x'$  and  $y'$ . Suppose that a point  $P$  has coordinates  $(x, y)$  with respect to the old system of  $x$  and  $y$  axes, and coordinates  $(x', y')$  with respect to the new axes. Then given  $\alpha$  and one pair of coordinates, the program allows you to find the other pair of coordinates.



### Equations:

Let Rect  $(r, \theta)$  denote the operation  $\boxed{R}$  when  $r$  is in the X-register and  $\theta$  is in the Y-register. Let Pol  $(x, y)$  denote the operation  $\boxed{P}$  when  $x$  is in X and  $y$  is in Y.

Then  $(x', y') = \text{Rect}(r, \theta - \alpha)$   
 where  $(r, \theta) = \text{Pol}(x - x_0, y - y_0)$

and  $(x, y) = (x_0, y_0) + \text{Rect}(r', \theta' + \alpha)$   
 where  $(r', \theta') = \text{Pol}(x', y')$

**Remarks:**

The program may be used to solve a problem of translation only, or of rotation only, or of combined translation and rotation. If the problem involves translation alone, a value of  $\alpha = 0$  must be input. For rotation alone, the values  $x_0 = y_0 = 0$  must be input.

The program assumes the following sign convention:  $\alpha$  should be input as a positive number if the rotation is counterclockwise, and negative if clockwise.

This program assumes the calculator is set in DEG mode.

A-000 LBLA	$x_0 \uparrow y_0 \uparrow \theta$	A-020 XZY	
A-001 S 7		A-021 PRTX	
A-002 R $\downarrow$		A-022 SPC	
A-003 S 9		A-023 RTN	
A-004 R $\downarrow$		A-024LBL2	
A-005 S 8		A-025 XZY	
A-006 RTN		A-026 +P	
A-007LBL1		A-027 XZY	
A-008 R 9		A-028 R 7	
A-009 -		A-029 +	
A-010 XZY		A-030 XZY	
A-011 R 8		A-031 +R	
A-012 -		A-032 R 8	
A-013 +P		A-033 +	
A-014 XZY		A-034 PRTX	
A-015 R 7		A-035 XZY	
A-016 -		A-036 R 9	
A-017 XZY		A-037 +	
A-018 +R		A-038 PRTX	
A-019 PRTX		A-039 SPC	
REGISTERS			
0	1	2	3
4	5	6	7 $\theta$
8 $x_0$	9 $y_0$	.0	.1
.2	.3	.4	.5 I

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in new origin and angle of rotation.	$x_0$	<b>ENTER</b>	
		$y_0$	<b>ENTER</b>	
		$\alpha$	<b>A</b>	$x_0$
3	Key in coordinates in old system and compute co-			

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	ordinates in the new.	x	<b>ENTER</b>	
		y	<b>GSB</b> <b>1</b>	x'
				y'
4	Key in coordinates in new system and compute coordinates in the old.	x' y'	<b>ENTER</b> <b>GSB</b> <b>2</b>	x y

**Example:**

The origin of a coordinate system is translated to the point  $(-1, 4)$  and rotated  $30^\circ$  in a positive (counterclockwise) direction. Find the new coordinates of the point whose coordinates in the old system are  $(1, 3)$ . If the coordinates of a point in the new system are  $(5, 7)$ , what are its coordinates in the old system?

**Keystrokes:**

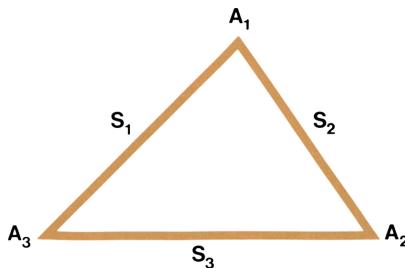
- 1 **CHS** **ENTER** 4 **ENTER** 30 **A** **▶**  
 1 **ENTER** 3 **GSB** **1**  $\longrightarrow$   
 5 **ENTER** 7 **GSB** **2**  $\longrightarrow$

**Outputs:**

- 1.00  
 1.23 \*\*\* (x')  
 -1.87 \*\*\* (y')  
 -0.17 \*\*\* (x)  
 12.56 \*\*\* (y)

## TRIANGLE SOLUTIONS

This program may be used to find the sides, the angles, and the area of a plane triangle.



In general the specifications of any three of the six parameters of a triangle (3 sides, 3 angles) is sufficient to define the triangle. (The exception is that the three angles will not define a triangle.) There are thus five possible cases that this program will handle: two sides and the included angle (SAS), two angles and the included side (ASA), two sides and the adjacent angle (SSA—the ambiguous case), two angles and the adjacent side (AAS), and three sides (SSS).

The calculator will successively print the values of the sides, the angles, and the area. The order of output is determined by the order of input: if input values are selected in a clockwise order, output values will be printed in a clockwise order. The order of outputs is as follows:

First side input ( $S_1$ )

Adjacent angle ( $A_1$ )

Adjacent side ( $S_2$ )

Adjacent angle ( $A_2$ )

Adjacent side ( $S_3$ )

Adjacent angle ( $A_3$ )

Area

**Equations:**

**SAS** ( $S_1$ ,  $A_1$ ,  $S_2$ ):

$$S_3 = \sqrt{S_1^2 + S_2^2 - 2 S_1 S_2 \cos A_1}$$

$$A_2 = \tan^{-1} \frac{S_1 \sin A_1}{S_2 - S_1 \cos A_1}$$

$$A_3 = \cos^{-1} [-\cos(A_1 + A_2)]$$

**ASA** ( $A_3, S_1, A_1$ ):

$$S_2 = S_1 \frac{\sin A_3}{\sin A_2} = S_1 \frac{\sin A_3}{\sin(A_1 + A_3)}$$

Now go to SAS.

**SSA** ( $S_1, S_2, A_2$ ):

$$A_3 = \sin^{-1} \left( \frac{S_2 \sin A_2}{S_1} \right)$$

$$A_1 = \cos^{-1} [-\cos(A_2 + A_3)]$$

Now go to SAS.

**AAS** ( $A_2, A_1, S_1$ ):

$$S_2 = S_1 \frac{\sin A_3}{\sin A_2} = S_1 \frac{\sin(A_1 + A_2)}{\sin A_2}$$

Now go to SAS.

**SSS** ( $S_1, S_2, S_3$ ):

$$A^1 = \cos^{-1} \left( \frac{S_1^2 + S_2^2 - S_3^2}{2 S_1 S_2} \right)$$

Now go to SAS.

### Remarks:

Any angular mode may be used.

Note that the triangle described by the program does not conform to standard triangle notation; i.e.,  $A_1$  is not opposite  $S_1$ .

Angles must be entered as decimals. The **H.MS→D** conversion can be used to convert degrees, minutes, and seconds to decimal degrees.

Accuracy of solution may degenerate for triangles containing extremely small angles.

A-000 LBLA		A-060 R 3	
A-001 S 3	SAS	A-061 x	
A-002 R↓		A-062 2	
A-003 S 2		A-063 x	
A-004 R↓		A-064 ÷	
A-005 S 1		A-065 COS⁻¹	
A-006 LBL9		A-066 S 2	
A-007 R 2	Compute A <sub>2</sub> , S <sub>3</sub> by law of cosines.	A-067 GTD9	
A-008 R 1		B-000 LBLB	
A-009 →R		B-001 S 2	
A-010 R 3		B-002 R↓	
A-011 X↔Y		B-003 S 1	
A-012 -		B-004 X↔Y	ASA
A-013 →P		B-005 S 6	
A-014 S 5		B-006 SIN	
A-015 X↔Y		B-007 R 2	
A-016 S 4		B-008 R 6	
A-017 R 2		B-009 +	
A-018 +		B-010 SIN	
A-019 COS	cos A <sub>3</sub> = -cos (A <sub>1</sub> + A <sub>2</sub> )	B-011 ÷	
A-020 CHS		B-012 x	
A-021 COS⁻¹		B-013 S 3	
A-022 S 6		B-014 JPA9	
A-023 SIN		C-000 LBLC	
A-024 x		C-001 S 4	
A-025 R 1		C-002 R↓	
A-026 x		C-003 S 3	
A-027 ²	Area	C-004 X↔Y	
A-028 ÷		C-005 S 1	
A-029 S 0		C-006 R 4	
A-030 R 1		C-007 SIN	
A-031 PRTX		C-008 X↔Y	
A-032 R 2	Output solutions.	C-009 ÷	
A-033 PRTX		C-010 x	
A-034 R 3		C-011 SIN⁻¹	
A-035 PRTX		C-012 R 4	
A-036 R 4		C-013 +	
A-037 PRTX		C-014 COS	
A-038 R 5		C-015 CHS	
A-039 PRTX		C-016 COS⁻¹	
A-040 R 6		C-017 S 2	
A-041 PRTX		C-018 GSB1	
A-042 SPC		C-019 R 1	
A-043 R 0		C-020 R 3	
A-044 PRTX		C-021 X↔Y	Two solutions exist if S <sub>2</sub> > S <sub>1</sub> .
A-045 SPC		C-022 RTN	
A-046 RTN		C-023 R 6	
A-047 LBL1		C-024 COS	
A-048 S 5	SSS	C-025 CHS	
A-049 R↓		C-026 COS⁻¹	
A-050 S 3		C-027 S 6	
A-051 R↓		C-028 R 4	
A-052 S 1	Find A <sub>1</sub> by law of cosines, go to SAS.	C-029 +	
A-053 R 3		C-030 COS	
A-054 →P		C-031 CHS	
A-055 X²		C-032 COS⁻¹	
A-056 R 5		C-033 S 2	
A-057 X²		C-034 LBL1	
A-058 -		C-035 JPA9	
A-059 R 1		D-000 LBLD	

$$A_3 \leftarrow 180 - A_3.$$

Find A<sub>1</sub>, go to SAS.

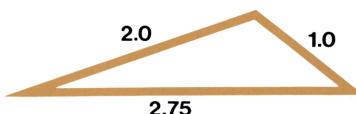
D-001 S 1 D-002 R↓ D-003 S 2 D-004 R↓ D-005 S 4 D-006 R↑ D-007 + D-008 SIN	AAS  Find $S_2$ , go to SAS.	D-009 R 4 D-010 SIN D-011 ÷ D-012 R 1 D-013 × D-014 S 3 D-015 JPA9	
<b>REGISTERS</b>			
0 Area	1 $S_1$	2 $A_1$	3 $S_2$
4 $A_2$	5 $S_3$	6 $A_3$	7
8	9	.0	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Select case.			
	<b>SAS</b>			
3	Key in side, angle, and side.	S	<b>ENTER</b>	
		A	<b>ENTER</b>	
		S	<b>A</b>	Output
	<b>ASA</b>			
4	Key in angle, side, and angle.	A	<b>ENTER</b>	
		S	<b>ENTER</b>	
		A	<b>B</b>	Output
	<b>SSA</b>			
5	Key in side, side, and angle.	S	<b>ENTER</b>	
		S	<b>ENTER</b>	
		A	<b>C</b>	Output*
	<b>AAS</b>			
6	Key in angle, angle, and side.	A	<b>ENTER</b>	
		A	<b>ENTER</b>	
		S	<b>D</b>	Output
	<b>SSS</b>			
7	Key in three sides.	S	<b>ENTER</b>	
		S	<b>ENTER</b>	

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
		S	g JUMP	
			A 1 R/S	Output
	Output in each case consists of the following variables in this order:			
	First side input			
	Adjacent angle			
	Adjacent side			
	Adjacent angle			
	Adjacent side			
	Adjacent angle			
	Area			
	*If two solutions exist, both will be printed.			

**Example 1:**

Find the angles and the area for the following triangle.

**Keystrokes:**

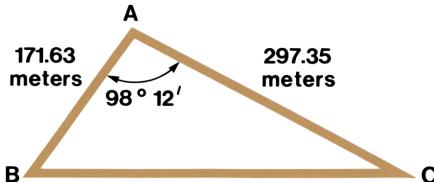
2 **ENTER** 1 **ENTER** 2.75 g **JUMP**  
**A 1 R/S** →

**Outputs:**

2.00 \*\*\*  
 129.84 \*\*\* (A<sub>1</sub>)  
 1.00 \*\*\*  
 33.95 \*\*\* (A<sub>2</sub>)  
 2.75 \*\*\*  
 16.21 \*\*\* (A<sub>3</sub>)  
 0.77 \*\*\* (Area)

**Example 2:**

A surveyor is to find the area and dimensions of a triangular land parcel. From point A, the distances to B and C are measured with an electronic distance meter. The angle between AB and AC is also measured. Find the area and other dimensions of the triangle.



This is a side-angle-side problem where:

$$S_1 = 171.63, A_1 = 98^\circ 12' \text{ and } S_2 = 297.35.$$

**Keystrokes:**

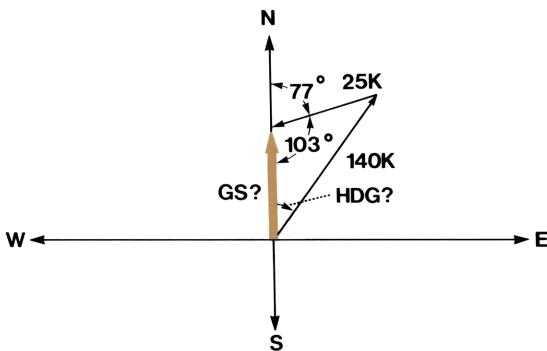
171.63 **ENTER** 98.12 **F1** **H.M.S↔H**  
297.35 **A** →

**Outputs:**

171.63 \*\*\* (AB)  
98.20 \*\*\* (A)  
297.35 \*\*\* (AC)  
27.83 \*\*\* (C)  
363.91 \*\*\* (CB)  
53.97 \*\*\* (B)  
25256.21 \*\*\* (Area)

**Example 3:**

A pilot wishes to fly due north. The wind is reported as 25 knots at  $77^\circ$ . Because winds are reported opposite to the direction they blow, this is interpreted as  $77 + 180$  or  $257^\circ$ . The true airspeed of the aircraft is 140 knots. What heading (HDG) should be flown? What is the ground speed (GS)?



By subtracting the wind direction from 180 (yielding an angle of  $103^\circ$ ), the problem reduces to a  $S_1, S_2, SA_2$  triangle.

#### Keystrokes:

140 [ENTER] 25 [ENTER] 103

**C** →

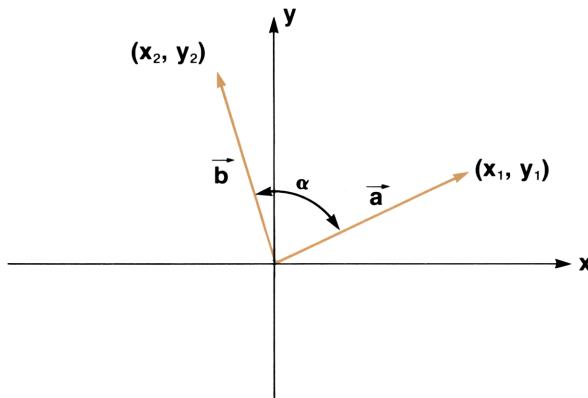
#### Outputs:

140.00 \*\*\* (TAS)  
 66.98 \*\*\*  
 25.00 \*\*\* (Wind velocity)  
 103.00 \*\*\*  
 132.24 \*\*\* (GS)  
 10.02 \*\*\* (HDG)  
 1610.64 \*\*\*

Thus, the pilot should fly a heading  $10.02^\circ$  east of due north. His ground speed equals 132.24 knots.

## VECTOR OPERATIONS

This program performs some of the most common operations on two-dimensional vectors, including addition, subtraction, dot product, and angle between the vectors.



Vectors may be input in either polar or rectangular form. The two vectors input need not be in the same form. Output vectors are printed in rectangular coordinates; pressing [R/S] will cause them to be printed in polar coordinates as well.

**Equations:**

Let the vectors  $\vec{a}$  and  $\vec{b}$  be  $(x_1, y_1)$  and  $(x_2, y_2)$  in rectangular coordinates or  $(r_1, \theta_1)$  and  $(r_2, \theta_2)$  in polar coordinates. Let  $\alpha$  be the angle between  $\vec{a}$  and  $\vec{b}$ .

Then

$$\vec{a} + \vec{b} = (x_1 + x_2, y_1 + y_2)$$

$$\vec{a} - \vec{b} = (x_1 - x_2, y_1 - y_2)$$

$$\vec{a} \cdot \vec{b} = x_1 x_2 + y_1 y_2$$

$$\alpha = \cos^{-1} \left( \frac{\vec{a} \cdot \vec{b}}{r_1 r_2} \right)$$

**Remarks:**

This program assumes the calculator is set in DEG mode.

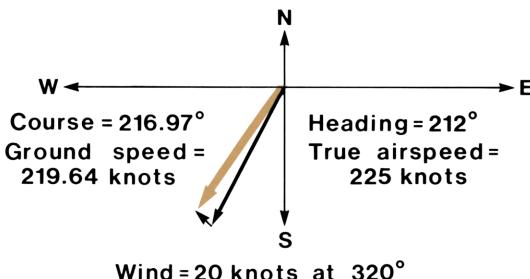
A-000 LBLA A-001 R 7 A-002 S 9 A-003 R↓ A-004 S 7 A-005 R↓ A-006 R 6 A-007 S 8 A-008 R↓ A-009 S 6 A-010 RTN A-011 LBL1 A-012 R 6 A-013 R 8 A-014 + A-015 PRTX A-016 R 7 A-017 R 9 A-018 + A-019 PRTX A-020 SPC A-021 GT09 A-022 LBL2 A-023 R 8 A-024 R 6 A-025 - A-026 PRTX A-027 R 9 A-028 R 7 A-029 - A-030 PRTX A-031 SPC A-032 LBL9 A-033 R/S A-034 X↔Y A-035 →P A-036 PRTX	Input (x, y). ----- Addition. ----- Subtraction. ----- Convert from rectangular to polar.	A-037 X↔Y A-038 PRTX A-039 SPC A-040 RTN A-041 LBL3 A-042 GSB8 A-043 PRTX A-044 SPC A-045 RTN A-046 LBL4 A-047 GSB8 A-048 R 9 A-049 R 8 A-050 →P A-051 X↔Y A-052 R↓ A-053 R 7 A-054 R 6 A-055 →P A-056 X↔Y A-057 R↓ A-058 × A-059 ÷ A-060 COS⁻¹ A-061 PRTX A-062 SPC A-063 RTN A-064 LBL8 A-065 R 6 A-066 R 8 A-067 × A-068 R 7 A-069 R 9 A-070 × A-071 + B-000 LBLB B-001 X↔Y	----- Dot product. ----- Angle between. ----- Subroutine for dot product called from LBL 3 and LBL 4.
---	---	--	--

<b>B-002 <math>\rightarrow R</math></b>	Input $(r, \theta)$ , convert to $(x, y)$ .		<b>B-004 GTOA</b>	-----
<b>REGISTERS</b>				
0	1	2	3	
4	5	6	7	
8 $x_1$	9 $y_1$	.0	.1	
.2	.3	.4	.5	1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Key in vector $\vec{a}$ :			
	● Rectangular	$x_1$	<b>ENTER</b>	
		$y_1$	<b>A</b>	$x_1$
	● Polar	$r_1$	<b>ENTER</b>	
		$\theta_1$	<b>B</b>	$x_1$
3	Key in vector $\vec{b}$ :			
	● Rectangular	$x_2$	<b>ENTER</b>	
		$y_2$	<b>A</b>	$x_2$
	● Polar	$r_2$	<b>ENTER</b>	
		$\theta_2$	<b>B</b>	$x_2$
4	Compute:			
	● Sum $(\vec{a} + \vec{b})$		<b>GSB [1]</b>	$x$
				$y$
	● Difference $(\vec{a} - \vec{b})$		<b>GSB [2]</b>	$x$
				$y$
	● Dot product $(\vec{a} \cdot \vec{b})$		<b>GSB [3]</b>	$\vec{a} \cdot \vec{b}$
	● Angle between		<b>GSB [4]</b>	$\alpha$

**Example 1:**

An aircraft flies a heading of 212 degrees at 225 knots. The wind blows at 20 knots from 140 degrees, i.e., towards 320 degrees. What is the course and ground speed of the aircraft?

**Keystrokes:**

225 [ENTER] 212 [B] →  
 20 [ENTER] 320 [B] →  
 GSB [1] →  
 R/S →

**Outputs:**

-190.81  
 15.32  
 -175.49 \*\*\* (x)  
 -132.09 \*\*\* (y)  
 219.64 \*\*\* (Knots)  
 -143.03 \*\*\* (Degrees)

**Example 2:**

Find the dot product of and the angle between the vectors  $(2, -5)$  and  $(4, 1)$ .

**Keystrokes:**

2 [ENTER] 5 CHS [A] →  
 4 [ENTER] 1 [A] →  
 GSB [3] →  
 GSB [4] →

**Outputs:**

2.00  
 4.00  
 3.00 \*\*\* (Dot product)  
 82.23 \*\*\* (Angle)

## HYPERBOLICS

This program computes the hyperbolic functions and their inverses. Note in the equations below the restrictions on the values of the argument in each case.

### **Equations:**

#### *Hyperbolic functions*

$$\sinh x = \frac{e^x - e^{-x}}{2}$$

$$\cosh x = \frac{e^x + e^{-x}}{2}$$

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\operatorname{csch} x = \frac{1}{\sinh x} \quad (x \neq 0)$$

$$\operatorname{sech} x = \frac{1}{\cosh x}$$

$$\coth x = \frac{1}{\tanh x} \quad (x \neq 0)$$

#### *Inverse Hyperbolic Functions*

$$\sinh^{-1} x = \ln [x + (x^2 + 1)^{1/2}]$$

$$\cosh^{-1} x = \ln [x + (x^2 - 1)^{1/2}] \quad x \geq 1$$

$$\tanh^{-1} x = \frac{1}{2} \ln \left[ \frac{1+x}{1-x} \right] \quad x^2 < 1$$

$$\operatorname{csch}^{-1} x = \sinh^{-1} \left[ \frac{1}{x} \right] \quad x \neq 0$$

$$\operatorname{sech}^{-1} x = \cosh^{-1} \left[ \frac{1}{x} \right] \quad 0 < x \leq 1$$

$$\coth^{-1} x = \tanh^{-1} \left[ \frac{1}{x} \right] \quad x^2 > 1$$

**Remarks:**

This program assumes the calculator is set in RAD mode.

A-000 LBL4 A-001 LBL1 A-002 e <sup>x</sup> A-003 ENT↑ A-004 1/X A-005 - A-006 2 A-007 ÷ A-008 RTN A-009 LBL2 A-010 e <sup>x</sup> A-011 ENT↑ A-012 1/X A-013 + A-014 2 A-015 ÷ A-016 RTN A-017 LBL3 A-018 S <sup>-1</sup> A-019 GSB1 A-020 R <sup>-1</sup> A-021 GSB2 A-022 ÷ A-023 RTN A-024 LBL4 A-025 ENT↑ A-026 X <sup>2</sup>		Sinh.	A-027 1 A-028 + A-029 √X A-030 - A-031 LN A-032 RTN A-033 LBL5 A-034 ENT↑ A-035 X <sup>2</sup> A-036 1 A-037 - A-038 √X A-039 + A-040 LN A-041 RTN A-042 LBL6 A-043 1 A-044 X <sup>2</sup> Y A-045 + A-046 1 A-047 LSTX A-048 - A-049 ÷ A-050 LN A-051 2 A-052 ÷	Sinh <sup>-1</sup> Cosh <sup>-1</sup> Tanh <sup>-1</sup>
<b>REGISTERS</b>				
0	1	2	3	
4	5	6	7	
8	9	.0	.1	
.2	.3	.4	.5	.1

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	Key in program.			
2	Initialize.		GTO A	
3	For hyperbolics, go to step 4; for inverse hyperbolics, go to step 5.			
	<b>HYPERBOLIC FUNCTIONS</b>			
4	Key in argument and compute			
	• hyperbolic sine	x	GSB 1	sinh x
	• hyperbolic cosine	x	GSB 2	cosh x

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
	• hyperbolic tangent	x	GSB [3]	tanh x
	• hyperbolic cotangent	x	GSB [3] $\frac{1}{x}$	coth x
	• hyperbolic secant	x	GSB [2] $\frac{1}{x}$	sech x
	• hyperbolic cosecant	x	GSB [1] $\frac{1}{x}$	csch x
<b>INVERSE HYPERBOLIC FUNCTIONS</b>				
5	• inverse hyperbolic sine	x	GSB [4]	$\sinh^{-1} x$
	• inverse hyperbolic cosine	x	GSB [5]	$\cosh^{-1} x$
	• inverse hyperbolic tangent	x	GSB [6]	$\tanh^{-1} x$
	• inverse hyperbolic cotangent	x	$\frac{1}{x}$ GSB [6]	$\coth^{-1} x$
	• inverse hyperbolic secant	x	$\frac{1}{x}$ GSB [5]	$\operatorname{sech}^{-1} x$
	• inverse hyperbolic cosecant	x	$\frac{1}{x}$ GSB [4]	$\operatorname{csch}^{-1} x$

**Example 1:**

Evaluate the following hyperbolic functions:

 $\sinh 2.5$ ;  $\cosh 3.2$ ;  $\tanh 1.9$ ;  $\coth -2.01$ ;  $\operatorname{sech} -0.25$ ;  $\operatorname{csch} 4.6$ .**Keystrokes:**

GTO A

2.5 GSB [1] →  
 3.2 GSB [2] →  
 1.9 GSB [3] →  
 2.01 CHS GSB [3]  $\frac{1}{x}$  →  
 .25 CHS GSB [2]  $\frac{1}{x}$  →  
 4.6 GSB [1]  $\frac{1}{x}$  →

**Outputs**

6.05      ( $\sinh 2.5$ )  
 12.29     ( $\cosh 3.2$ )  
 0.96      ( $\tanh 1.9$ )  
 -1.04     ( $\coth -2.01$ )  
 0.97      ( $\operatorname{sech} -0.25$ )  
 0.02      ( $\operatorname{csch} 4.6$ )

**Example 2:**Evaluate the following inverse hyperbolic functions:  $\sinh^{-1} 2.4$ ;  $\cosh^{-1} 90$ ;  $\tanh^{-1} -0.65$ ;  $\coth^{-1} 3.4$ ;  $\operatorname{sech}^{-1} 0.4$ ;  $\operatorname{csch}^{-1} 2$ .**Keystrokes:**

GTO A

2.4 GSB [4] →

**Outputs:**1.61      ( $\sinh^{-1} 2.4$ )

90 GSB 5	→	5.19	$(\cosh^{-1} 90)$
.65 C GSB 8	→	-0.78	$(\tanh^{-1} -0.65)$
3.4 !x GSB 6	→	0.30	$(\coth^{-1} 3.4)$
.4 !x GSB 5	→	1.57	$(\operatorname{sech}^{-1} 0.4)$
2 !x GSB 4	→	0.48	$(\operatorname{csch}^{-1} 2)$



HEWLETT  PACKARD

1000 N.E. Circle Blvd., Corvallis, OR 97330

00095-90003